СРАВНЕНИЕ ПЛЕЙЕРЫ

БЕСПЛАТНЫЙ DVD
Все для Pi: NOOBS, Jasper, KaliLinux, PiMusicBox, RetroPie

LINUX FORMAT

Главное в мире Linux

Январь 2015 № 1 (191/192)

Контейнеры Docker
» Создавайте и автоматизируйте свои сервисы через виртуализацию

ХАКИНГ PI

Раскрываем подноготную могучего малыша — платы с Linux-ПК

» Штучки Minecraft
» Потоковое вещание
» Хостинг ownCloud
» Доступ к файлам через Samba
» Воспользуемся API Twitter



Умелец, учитель, художник
«Вокруг Arduino образовалось большое сообщество, а это и есть ключ к успеху»
Майкл Шайлоу о могуществе Arduino с. 38

Также в номере...



Музыка для профи
» Не разбираетесь в JACK? Будем знакомы. Аудио в Linux с. 48

Rescatux
Спасение утопших
» Rescatux востановит вашу систему после катастрофы

Cython
Python из C
» Непринужденно переходим с языка на язык внутри кода

Mathematica Pi
Откроем личико
» Распознавание образин

ПОДПИСНЫЕ ИНДЕКСЫ В КАТАЛОГАХ
Агентство «Роспечать» — 36343,
«Почта России» — 11932, «Пресса России» — 90959



4 607130 825167 15001





Что мы делаем

» Мы поддерживаем открытое сообщество, предоставляя источник информации и площадку для обмена мнениями.

» Мы помогаем всем читателям получить от Linux максимум пользы, публикуя статьи в разделе «Учебники»: здесь каждый найдет что-то по своему вкусу.

» Мы выпускаем весь код, появляющийся на страницах раздела «Учебники», по лицензии GNU GPLv3.

» Мы стремимся предоставлять точные, актуальные и непредвзятые сведения обо всем, что касается Linux и свободного ПО.

Corporate Patron Free Software Foundation

Кто мы

Мы спросили наших авторов: какой проект прошедшего года они назвали бы ну самым своим любимым?



Джонни Бидвелл
Если хоть сколько-нибудь верить пропаганде, чертовски интересен Maidsafe — распределенная экосистема с децентрализованной, анонимной платформой приложений, поощряемых собственной криптовалютой. Платформа открыта, и применяемое шифрование тщательно продумано. Любопытно оценить реакцию...



Эндрю Моллетт
Для меня всегда всех побеждает *Puppet*, причем с легкостью. Хоть сам проект не нов, он, тем не менее, поддерживает настройку таких инноваций, как *OpenStack*. Его легкость и простота в управлении системами промышленного уровня делает мой выбор послушным, как марионетка. (Правильно, это каламбур.)



Лес Паундер
За последние 12 месяцев самое пристальное мое внимание привлек Shrimping. Это плата Arduino всего за £5, и проект действительно захватывает, как только вы принимаетесь собирать Arduino и изучать функции всех его компонентов. Недавно я выяснил, что он работает со Scratch — НА ДЕЛЕ!



Ричард Смедли
Mailpile обеспечивает столь нужную защиту шифрованной почты пользователям-нетехнарям. Из толпы интересных проектов IndieWeb его выделяют мощное тэгирование, приметная быстрота и гибкость выбора хостинга. Вследствие чего *Mailpile* стал первой реалистичной альтернативой реально мощному Gmail.



Маянк Шарма
Похоже, стоит мне назвать какой-нибудь проект своим любимым, как он норовит тут же зачахнуть — SolusOS, Pear Linux и, из последних, Bodhi Linux. Ну и вы понимаете, я немного суеверен насчет этого слова. Хотя постойте... вот прямо сейчас, мой любимый проект — Windows!



Валентин Синицын
Это *Jailhouse*. Да, я отлично понимаю, что вы никогда о нем не слышали, и доброй половине из вас он едва ли когда-либо пригодится. Но я провел немало интересных часов за его отладкой, и думаю продолжить это занятие и в 2015 году.



Учебный снаряд

» Тему этого номера можно сформулировать так: «Нестандартные способы использования аппаратных возможностей Raspberry Pi». (Сказать «хакинг» — не по-русски, но много короче.) Некоторые из этих способов, например, вещание FM-радио через GPIO, действительно нетривиальны.

Такие «фокусы» стали возможны благодаря важнейшему, по утверждению авторов, решения: вывести «наружу» не только все три основных аппаратных интерфейса (последовательный порт, I2C и SPI), но и выводы цифрового сигнального процессора, размещенного на том же кристалле, и предоставить удобные библиотеки для работы с ними. Первое позволяет подключать практически любые датчики, а второе — формирвать весьма стабильные сигналы широтно-импульсной модуляции для управления электродвигателями с высочайшей, по любительским меркам, точностью. Проверял лично — до 1–2 граммов тяги воздушного винта. Вы уже догадались, зачем это нужно. Спасибо Энди Бэкеру [Andy Backer, www.themagpi.com/issue/issue-19/] за идею.

Возьму на себя смелость утверждать, что с поставленной задачей авторы Raspberry Pi справились блестяще. Создан великолепный, как говорили в эпоху парусного флота, «учебный снаряд» не только для упражнений в программировании, но и для освоения электроники и робототехники. Вот только *LibreOffice* на нем не работает — но это в задачу и не входило...

Кирилл Степанов
Главный редактор
» info@linuxformat.ru

Как с нами связаться

Письма для публикации: letters@linuxformat.ru
Подписка и предыдущие номера: subscribe@linuxformat.ru
Техническая поддержка: answers@linuxformat.ru
Общие вопросы: info@linuxformat.ru
Проблемы с дисками: disks@linuxformat.ru
Вопросы распространения: sales@linuxformat.ru
Сайт: www.linuxformat.ru, группа «ВКонтакте»: vk.com/linuxform
› **Адрес редакции:** Россия, Санкт-Петербург, пр. Медиков, 5, корп. 7
› **Телефон редакции:** (812) 309-0686. Дополнительная информация на с. 112



Содержание

«Будь добрым, ибо каждый твой встречный ведет более тяжкую битву» — Платон.



Обзоры

› **Кто сказал, что Ubuntu — не самая прекраснейшая ОС?**

Люди говорят



> Об Open Source я узнал прежде, чем это стало термином

На вашем бесплатном DVD

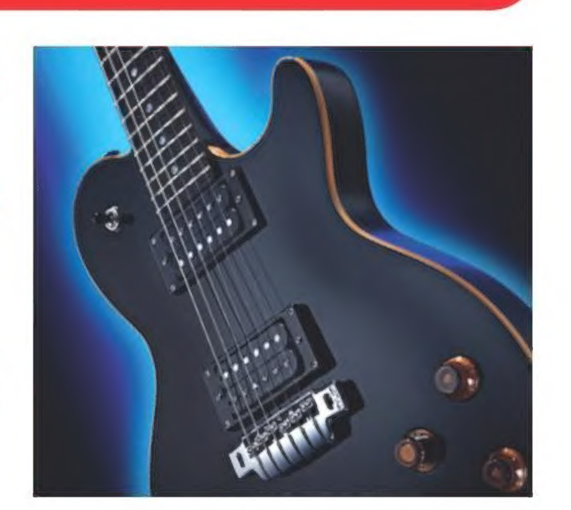


Raspbian, KaliLinux, PiMusicBox, RetroPie, Jasper 2014, NOOBS

» Все необходимое для Raspberry Pi

ПЛЮС: HotPicks и коды учебников с. 106

Ищите в этом номере



Академия кодинга



```
Map[rebuildFromEigenfaces[coefficients2, meanImage,
   eigenfaces, #] &, Range[4]]
```



Учебники



› Пингвины не дадут вам утонуть.

Постоянные рубрики



В ЭТОМ НОМЕРЕ: Суперкомпьютеры » Российский Gentoo » Linux ширится » Долой вирусы » Забудем трудности перевода » Роса и облака » Эти глаза напротив » ГИС оффлайн



» Рубрику готовил АНДРЕЙ ГОНДАРЕНКОВ

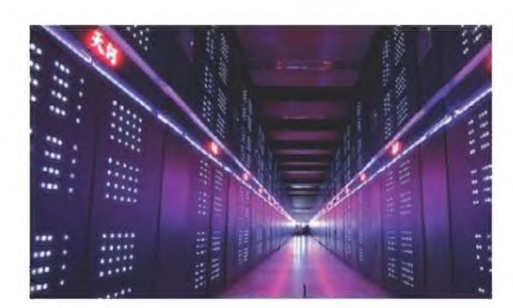
TOP-500

Вычислительная элита

Гонка суперкомпьютеров сбавляет обороты.



Согласно 44-му TOP-500 — рейтингу самых мощных суперкомпьютеров планеты — быстрейшей в мире системой в 4-й раз подряд оказался Тяньхэ — Tianhe-2 (Национальный университет оборонных технологий Китая). С июня 2013 года, когда Tianhe-2 впервые поднялся на первую строчку TOP-500, его производительность не менялась: 33,86 петафлоп/с в тесте Linpack. В первой десятке лишь один новичок: система от Cray для неназванного госучреждения США, её показатель — 3,57 петафлоп/с.



› Китайский кластер Tianhe-2, работающий под управлением Ubuntu Kylin, четвёртый раз подряд стал самым мощным суперкомпьютером планеты.

Учредители TOP-500 отмечают, что отсутствие конкуренции в верхней части списка тормозит его рост в целом, а среди сдерживающих факторов называют снижение госфинансирования исследований и предоставление провайдерами облачных сервисов типа Amazon и Microsoft масштабируемых суперкомпьютерных услуг, не требующих значительных авансовых капитальных затрат на установку гигантских машин в помещениях.

Суммарная мощность всех 500 систем нового рейтинга — 309 петафлоп/с (в июне было 274 петафлоп/с, а год назад — 250 петафлоп/с). Большинство их по-прежнему из США, хотя это число упало до исторического минимума — 231 (против 233 в июне 2014 и 265 в ноябре 2013). Присутствие в TOP-500 Азии снизилось до 120 (в июне было 132). Зато «европейцев» стало 130 (в июне — 116). Доминирует в поставке суперкомпьютерных систем Hewlett-Packard: 179 позиций, или около 36%. Далее идут IBM — 153 (30%) и Cray — 62 (12,4%). 485 систем (97%) из списка работают на GNU/Linux, 13 (2,6%) — на UNIX. Один суперкомпьютер работает под управлением смешанных ОС.

В новом TOP-500—9 суперкомпьютеров из России: T-Platform A-Class (МГУ) — 22-е место, Lomonosov — 58-е место, Tornado — 81-е место.

ДИСТРИБУТИВЫ

Calculate Linux

Под занавес уходящего года вышел релиз 14.12.

Этот развиваемый российской командой дистрибутив основан на Gentoo и предназначался для применения в корпоративной среде; ортодоксальные Gentoo'шники не считают его «кошерным» (классический Gentoo обязан быть скомпилирован и собран из исходных текстов на целевой машине), но многие опытные пользователи относят Calculate Linux к числу лучших систем, в том числе для дома. Основные изменения:

» оптимизирована утилита обновления системы (ускорена работа проверки обновлений в фоновом режиме; обновление более не выполняется отдельными этапами для утилит, библиотек Perl/Python и прочих пакетов; если в настраиваемом файле был изменён только заголовок, изменения будут приняты автоматически; добавлена поддержка чистки устаревших версий скачанных пакетов программ);

» введена поддержка установки *KVM* на Virtio-диски как гостевую систему;

» в утилите смены профиля *cl-update-profile* добавлен выбор способа обновления состава пакетов системы, добавлено обновление настроек системы при смене профиля;

» в Calculate Console обновлены иконки и упорядочено размещение служб;

» в CLD и CLDX включены обои нескольких последних версий;

» добавлена утилита настройки автовхода и шифрования директории пользователя;

» улучшена работа тачпада;

» обновлены часовые пояса;

» из состава CLDX исключён агрегатор RSS новостей *Liferea* в пользу плагина RSS в Claws-Mail;

» просмотрщик PDF документов в *Evince* в CLDX заменён на *QpdfView*;

» просмотрщик изображений *Eog* заменён на *Nomacs*;

» в CLDX добавлена утилита сведений о системе и бенчмарк HardInfo;

» исключён из образа менеджер фотографий *Shotwell*;

» в образ CLS включены библиотеки *FFmpeg*;

» для сокращения размера образа из состава CLS удален пакет обоев;

» утилиты управления частотой процессора *cpufrequtils* заменёны на *cpupower*.

Источник: www.computerworld.com; www.calculate-linux.ru

ПРОГРЕСС LINUX

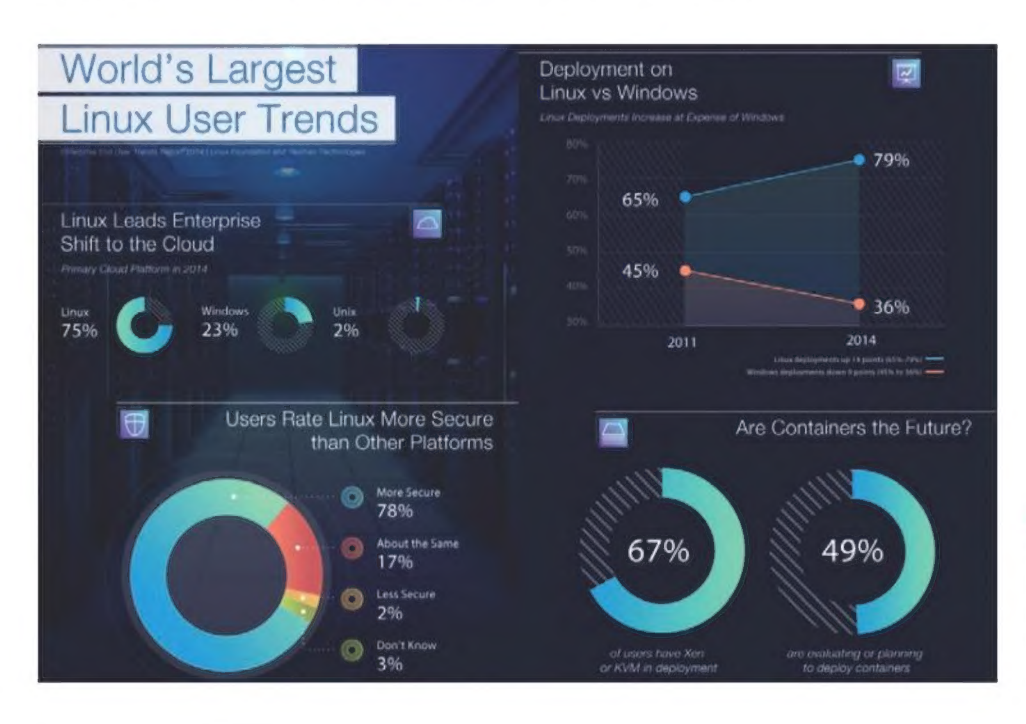
Linux шагает по облакам

ОС Linux энергично завоевывает корпоративные серверы.

Представленный Linux Foundation совместно с Yeoman Technology Group отчёт “2014 Enterprise End User Trends Report” свидетельствует, что в настоящее время GNU/Linux уверенно занимает место основной корпоративной «облачной» платформы. Пользователи считают эту операционную систему более безопасной, чем альтернативы. За прошедшие 4 года количество развертываний GNU/Linux увеличилось на 14%, в то время как аналогичный показатель Windows снизился на 9%. При составлении документа использовались данные, полученные от организаций, чьи объёмы продаж составляют $500 млн или более, или со штатом 500 или более сотрудников.

Ключевые положения “2014 Enterprise End User Trends Report”:

» 75% предприятий в рассматриваемой выборке используют GNU/Linux в качестве своей основной облачной платформы. Доля Windows составляет менее 24%, а UNIX — менее 2%;



› **Относительное число инсталляций Linux-систем выросло, а Windows-систем — продолжает снижаться.**

» в качестве корпоративной платформы GNU/Linux превосходит альтернативы техническими качествами, уровнем безопасности и стоимостью. 78% предприятий считают GNU/Linux наиболее безопасной операционной системой.

» в текущем году на 87% предприятий были добавлены серверы под управлением GNU/Linux, а 82% планируют развернуть такие серверы в 2015 году;

» стремительный рост популярности GNU/Linux вызывает ощутимую нехватку подготовленных специалистов: более 40% крупнейших предприятий обеспокоены поиском персонала, обученного работе с GNU/Linux (и этот пункт является самой серьёзной из перечисленных в отчёте проблем).

Аманда Макферсон [Amanda McPherson], вице-президент Linux Foundation по маркетингу и программам разработки, отмечает, что современный подход к созданию программного обеспечения смоделирован на основе методологии и принципов, использованных при создании Linux — крупнейшего в мире совместно разрабатываемого проекта, а нынешний отчёт даёт важную информацию об использовании корпоративными пользователями GNU/Linux в самых сложных средах.

СКАЖЕМ ВИРУСАМ «НЕТ»

Нужен ли Linux антивирус?

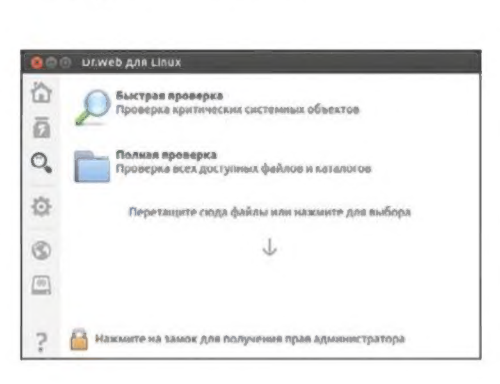
Компания «Доктор Веб» выпустила продукт Dr.Web версии 10.0 для Linux.

В новой версии добавлен web-антивирус, расширен перечень поддерживаемых дистрибутивов и улучшена работа продукта, ставшего еще более надежным и удобным в использовании средством защиты ПК под управлением Linux. Новый компонент — web-антивирус SpIDer Gate — значительно повышает уровень защиты ПК. Он осуществляет проверку входящего HTTP-трафика и блокирует передачу вредоносных объектов. Также этот компонент выполняет функции Офисного контроля, блокируя доступ к нежелательным ресурсам, что позволяет не только избежать интернет-угроз, но и обеспечить более эффективное использование рабочего времени.

Были внесены изменения в файловый монитор SpIDer Guard, что позволило улучшить механизм сканирования запущенных процессов для обезвреживания активных угроз, в том числе представляющих опасность для ОС Windows, запускаемых через *Wine*.

Продукт теперь поддерживает дистрибутивы Red Hat Enterprise Linux 7, 6.5 и 5.10, а также CentOS Linux версий 7, 6.5 и 5.10.

Вместе с тем была произведена оптимизация поддержки различных рабочих столов системой графической и звуковой нотификации событий, улучшена поддержка Dr.Web Enterprise Suite версии 10.0 и добавлена возможность единовременного запуска нескольких пользовательских сессий сканирования.



› **Dr.Web 10.0 для Linux теперь поддерживает дистрибутивы RHEL 7, 6.5 и 5.10, а также CentOS Linux версий 7, 6.5 и 5.10.**

Источник: www.linuxfoundation.org; news.drweb.com

МАШИННЫЙ ПЕРЕВОД

Translation Server 10

Компания PROMT анонсирует бизнес-переводчик под Linux.

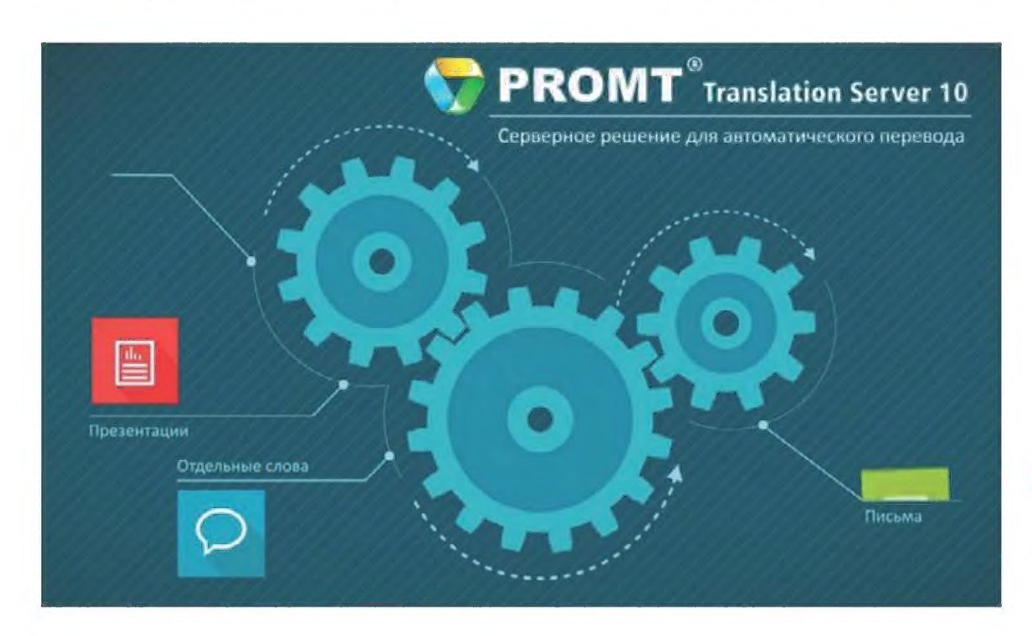
PROMT объявляет о выпуске корпоративного решения для перевода на платформе GNU/Linux. Переводчик создан на базе флагманского решения PROMT Translation Server.

PROMT Translation Server — надежное и масштабируемое решение, которое позволяет автоматизировать перевод любой информации. Специализированный продукт может быть установлен на 32- и 64-bit версии ОС Linux (включая Red Hat Linux). Первая версия поддерживает европейские языки (английский, испанский, итальянский, немецкий, португальский, французский). Доступен перевод документов в форматах DOCX, XLSX, PPTX, PDF. В наступившем году появится перевод на другие языки (например, казахский и финский), а также поддержка большего количества форматов.

> «Качественные решения для пользователей разных платформ.»

«Мы считаем, что переводчик под Linux будет интересен российскому бизнесу и государственным компаниям, — заявил Никита Шаблыков, руководитель отдела продаж PROMT. — Мировая практика развертывания серверов на Linux все больше распространяется и в России, особенно сейчас, когда актуальными стали темы импортозамещения и продвижения отечественного ПО. PROMT — российский разработчик с огромным опытом, и мы готовы предложить качественные решения для пользователей разных платформ».

С помощью PROMT Translation Server можно вести двуязычный документооборот, переводить деловую переписку, аналитические статьи и исследования из профильных журналов на иностранном языке, изучать законодательство других стран, стандарты, нормативы и многое другое. Важное преимущество продукта — гарантия безопасности: тексты переводятся на сервере компании, недоступном для третьих лиц. Система прав допускает независимую работу как отдельных пользователей, так и их групп.



› PROMT Translation Server 10 — основное решение для перевода в корпоративной сети.

ОБЛАКА ДЛЯ РОССИИ

На основе OpenStack

Разработано первое облачное решение на базе российской ОС.

Специалисты департамента сетевой интеграции компании ЛАНИТ разработали и собрали облачное решение на основе программного комплекса OpenStack и российских дистрибутивов GNU/Linux. Созданный продукт предназначен в первую очередь для внутреннего рынка и призван снизить зависимость потребителей от зарубежных вендоров и импортного программного обеспечения.

Облачное решение ЛАНИТ включает программный комплекс OpenStack, клиентские рабочие места и набор готовых преднастроенных сервисов на операционной системе РОСА. По сути, это самые популярные рабочие инструменты, которые используются в любой отрасли: сетевой каталог, файловый сервер, электронная почта, средства организации совместной работы, прокси-сервер и многое другое.

> «Все сервисы могут быть развернуты автоматически.»

Все эти сервисы могут быть развернуты в облаке автоматически и в любой момент предоставлены пользователю.

Внедряя продукты OpenStack, специалисты ЛАНИТ использовали компетенции технологических партнеров — компании Mirantis, которая входит в тройку лидеров ассоциации OpenStack и имеет богатый опыт построения публичных и приватных облаков. В решении задействован Mirantis OpenStack, программный продукт со свободным кодом — в его состав включен компонент FUEL, который позволяет быстро в автоматическом режиме развернуть облачную платформу.

Разработанное решение протестировано на совместимость с операционными системами ROSA «Enterprise Linux Server» и РОСА «Кобальт».

Созданный продукт впервые был представлен публике на CNews Forum 2014, который проводился 12 ноября 2014 года в Москве.

Источник: www.promt.ru; www.lanit.ru

ДЛЯ ВИДЕОКОНФЕРЕНЦИЙ

Бизнес на созвоне

Видеоконференцсвязь предлагается бесплатно.

Российская компания TrueConf выпустила новый продукт — TrueConf Server Free. Это корпоративное решение представляет собой уже зарекомендовавший себя TrueConf Server, распространяемый в рамках модели "Freemium". TrueConf Server Free — это полностью бесплатная платформа унифицированных коммуникаций, которая работает как в локальных сетях предприятий любой сложности, так и через Интернет.

Преимущества TrueConf Server Free:

» **Бесплатный сервер ВКС** Доступны все виды групповых видеоконференций до 6 участников включительно в разрешении вплоть до 4К (UltraHD).

» **Бесплатное скачивание** Клиентские приложения для видеоконференцсвязи от TrueConf можно скачать под используемую операционную систему (Windows, Linux, OS X, Android, iOS).

» **Инструменты для совместной работы** Адресная книга со статусами и фотографиями, обмен сообщениями, электронная доска, запись, передача файлов.

» **Масштабируемость** Благодаря SVC каждый участник получает максимально возможное качество видео и звука для своего устройства, канала связи, разрешения экрана и выбранной раскладки видеоокон.

» **WebRTC-конференции** Чтобы стать полноценными участниками с видео, чатом и возможностями совместной работы, достаточно перейти по ссылке через браузер.

» **Интеграция** с существующим VoIP и ВКС оборудованием для полноценного участия SIP-абонентов в видеоконференциях, и в том числе для вызовов на ТфОП.

» **Гибкая система лицензирования** для увеличения количества пользователей, позволяющая в любой момент перейти на расширенную версию TrueConf Server.

ГОРОЖАНАМ

Новый 2ГИС

Появилась версия для компьютеров и ноутбуков.

2ГИС представил бета-версию приложения для ноутбуков и компьютеров. Внешне она похожа на 2gis.ru, но работает оффлайн и имеет варианты для трех операционных систем — впервые для OS X и GNU/Linux, а также для Windows, как предыдущая версия.

Несмотря на повсеместное распространение Интернета, ситуации, когда его нет, остаются. Медленное соединение, ограниченный доступ к Wi-Fi или полное отсутствие интернет-подключения не помешают найти места и выбрать организации. В новом приложении основные функции работают без подключения к Сети. Интернет нужен только для обновления данных, которые будут подгружаться автоматически в фоновом режиме.

Приложение имеет такой же интерфейс, как и 2gis.ru.

«За год новый 2gis.ru показал, что мы движемся в правильном направлении. Нашим пользователям понравились его внешний вид и возможности. Теперь они доступны в новом интерфейсе приложения для компьютеров. Работать с ним можно так же, как с 2gis.ru, только без Интернета», заявил Павел Мочалкин, директор департамента новых продуктов 2ГИС.

Говорит Фёдор Кухарёнок, руководитель отдела внешних оффлайн-продуктов 2ГИС: "Прошлая версия не поддерживала динамичные изменения. Желание добавить новый тип данных — какую-нибудь информацию о городе — постоянно упиралось в дополнительные сроки разработки. Новый 2ГИС изначально проектировался так, чтобы такие изменения можно было делать без релиза новой версии. От реляционных баз мы отказались в пользу объектно-ориентированной. Раньше для получения информации об объекте приходилось делать селект из множества таблиц. Эта логика была зашита в приложение. В текущей версии мы полностью ушли от типизации объектов. Объект на момент рендеринга подбирается по данным, которые находятся в JSON'е». LXF

Новости короткой строкой

» Возможности онлайн-сервиса VirusTotal по предоставлению информации о вредоносном ПО расширены на платформу GNU/Linux.
Источник: www.zdnet.com

» В *ClamAV 0.98.5*, новом выпуске свободного антивирусного пакета, реализована поддержка формата XDP.
Источник: blog.clamav.net

» В рамках программы по передаче технологий АНБ США открыло исходный код и передает фонду Apache разработку проекта Niagarafiles (NiFi) — автоматизации распределения потоков данных в компьютерных сетях.
Источник: www.nsa.gov

» Для вывода из стагнации проекта Mozilla *Thunderbird* разработчики передают управление сообществу.
Источник: blog.mozilla.org

» Разработчики, не согласные со вводом в Debian GNU/Linux системы инициализации *systemd*, ответвили дистрибутив Devuan — без *systemd*.
Источник: lists.dyne.org

» Компания 2K Games, издатель стрелялки от первого лица *BioShock Infinite*, выпустит в 2015 г. Linux-версию.
Источник: www.phoronix.com

» Релиз ядра Linux 3.18 ввел интеграцию файловой системы OverlayFS и поддержку вывода звука через DisplayPort драйвером Nouveau, ускорил переход в спящий режим для многопроцессорных систем, ускорил сетевую подсистему при пакетной передаче данных.
Источник: lkml.org

» Драйвере AMD Catalyst 14.12 для Linux реализовал поддержку OpenCL 2.0 (на системах 64-bit с адаптерами AMD Radeon R) и декодирование через VAAPI (H264, VC1, MPEG2, MPEG4).
Источник: support.amd.com

» В *Darktable 1.6*, свободном фоторедакторе, реализованы поддержка мониторов с высоким разрешением и изображений большого размера, а также воспроизведение аудиозаметок.
Источник: www.darktable.org/news/

» Сетевой конфигуратор *NetworkManager 1.0* ввёл встроенный клиент DHCP, клиентскую библиотеку *libnm* и поддержку протокола DUN для выхода в сеть через Bluetooth (только для стека Bluez 5.x), а также расширил возможности IPv6.
Источник: mail.gnome.org

Источник: blog.trueconf.ru; info.2gis.ru

LINUX FORMAT Обзоры

Новинки программного и аппаратного обеспечения в описании наших экспертов



АЛЕКСЕЙ ФЕДОРЧУК
Тэг <сарказм> по умолчанию, смайлики по вкусу.

Релизы с ласковыми именами

Минувшая осень ознаменовалась двумя эпохальными событиями: сначала — выходом релиза среды Cinnamon 2.4, а затем — релиза Mint 17.1 Rebecca, где Cinnamon используется как одна из двух главных рабочих сред. Чего ж тут эпохального? — вправе вы спросить. Подумаешь, очередная версия очередной рабочей среды, а уж «вмещающий» её дистрибутив — вообще минорный релиз. Попробую ответить.

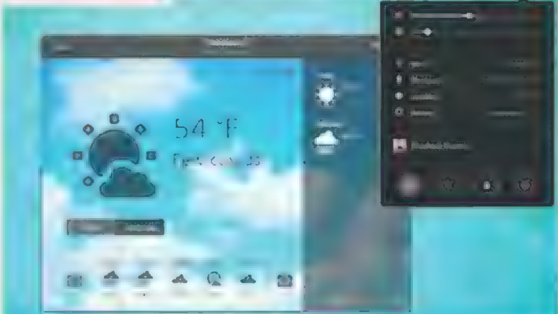
Версия Cinnamon 2.4 — последний шаг в превращении «гадкого утёнка» в прекрасного лебедя. А сцепка её с Mint придала системе законченность и гармонию: ни прибавить, ни убавить нечего. Кроме одного. С самого своего появления среда Cinnamon не отличалась, мягко говоря, бережным отношением к аппаратным ресурсам. Что исключало из числа её потенциальных применителей владельцев даже не старых и слабых машин, а среднеофисных конфигураций. И так было вплоть до Mint 17 с Cinnamon 2.2. А вот в релизах, о которых идёт речь, началась, как сказал бы Великий Мао, «борьба за упорядочивание стилей работы». То есть — за оптимизацию «железных» потребностей. И есть основание предполагать, что борьба эта будет продолжена в следующих промежуточных релизах Cinnamon-редакции Mint.

PS Заглавием я обязан Станиславу Шрамко aka stanis, напомнившему мне о книжке про похождения Эмиля Боева. А словами про гадкого утёнка и лебедя — Brego, автору сайта о Mint и Cinnamon: mintmem.com.

alv@posix.ru

Сегодня мы рассматриваем:

Gnome 3.14



› Чтобы узнать погоду, пользователям нового Gnome не обязательно выходить на улицу.

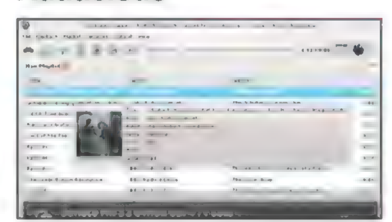
The Witcher 2



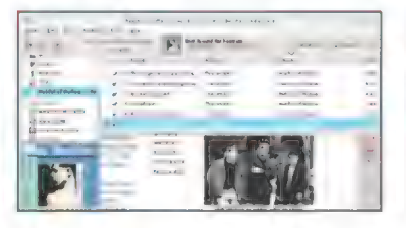
› Графика *The Witcher 2* отличается прямо-таки тициановской щедростью.

Сравнение: Музыкальные плейеры с. 24

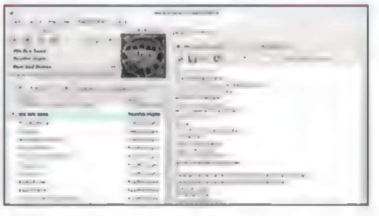
Audacious



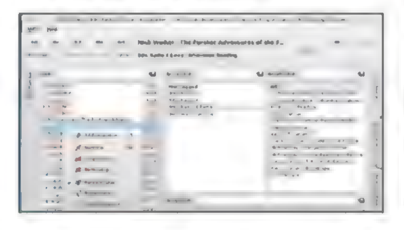
Banshee



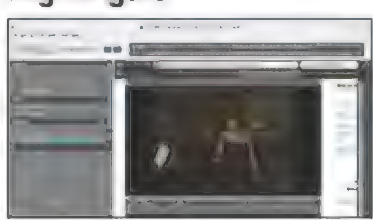
GMusicBrowser



MPD/QMPDClient



Nightingale



Приятно работать в сопровождении негромкой фоновой музычки — или, наоборот, ритмичной и заводной. Но не рухнет ли ваш плейер под тяжестью огромной фонотеки? Рассматриваем особо дюжие.

Gentoo Live 2014

Хотя Gentoo вроде не ходил ложными или слишком заурядными путями, **Шашанк Шарма** заметил: попытка превзойти совершенство оказалась наказуема...

Вкратце

» **Gentoo Live DVD до отказа набит приложениями на вкус любого пользователя. Это не устанавливаемый дистрибутив, но и для введения пользователя в Gentoo он тоже не предназначен.**

Gentoo славится тем, что раскалывает сообщество: одни от него в полном восторге, другие его боятся. Это сильно контрастирует с отношением к большинству других дистрибутивов: тех просто любят либо нет. Gentoo Live DVD — это, несмотря на название и происхождение, другой дистрибутив.

В отличие от большинства Live-дистрибутивов, его нельзя установить. Поэтому сравнения с тёзками не совсем корректны. Этот DVD не пытается привлечь к Gentoo новых пользователей. Он не является демонстрацией технологий Gentoo в надежде облегчить им привыкание. Gentoo DVD — это продукт сообщества Gentoo при содействии его разработчиков.

DVD предлагается в двух версиях: livedvd-x86-amd64-32ul-20140826 предназначена для 32-битной (x86) и 64-битной (x86_64) архитектур, тогда как livedvd-amd64-multilib-20140826 будет работать только на x86_64. Хотя нынешний релиз DVD значительно «легче» по сравнению с предыдущим, он по-прежнему до краёв наполнен лучшими приложениями, предлагаемыми экосистемой Linux.

Проверим комплектацию

Самое замечательное в DVD то, что в нём есть всё для демонстрации возможностей Linux новым заинтересованным пользователям. Речь не только о приложениях: для полноты представления, DVD включает целый ряд рабочих столов. Он основан на ядре Linux 3.15.6, а по части рабочих столов предлагает KDE, Gnome, *XFCE*, *Fluxbox*, *LXQT*, а также *i3* — мозаичный менеджер окон, ориентированный на разработчиков и продвинутых пользователей.



› **Выпуск 20140826, как можно бы и догадаться по малоудобному названию, является расширяемым, но непримечательным дистрибутивом Linux, и ничем больше.**

Если при загрузке с USB-носителя вы хотите использовать постоянное хранилище данных, нужно создать EXT-раздел для файлов. Мы использовали *fdisk* для создания раздела и команду **mkfs.ext3** для создания файловой системы. При следующей загрузке в Gentoo выберите образ, который хотите вызвать, и нажмите F2. Затем наберите **aufs=<устройство>** в строке параметров ядра [у меня в *VirtualBox* не так: вместо F2 я нажимаю Tab, и тогда могу набирать строку параметров ядра. Из F2 можно только дойти до быстрой справки, где это и написано, — *прим. пер.*]. Это придется делать при каждой загрузке в Gentoo или смене компьютера. При загрузке DVD в приглашении *Grub* жмите на функциональные клавиши, чтобы получать инструкции по различным параметрам загрузки.

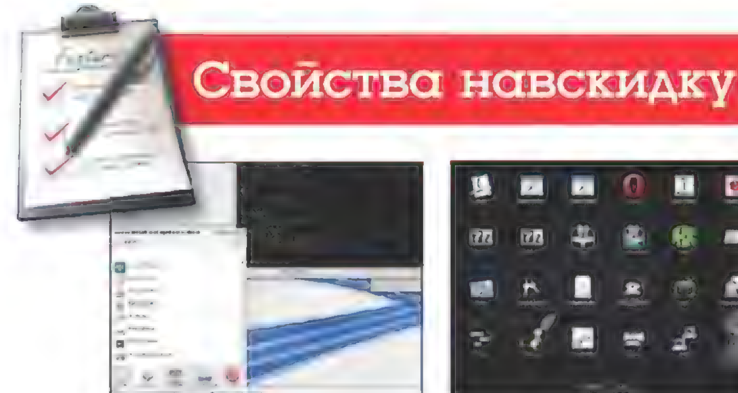
На сайте проекта ведётся список всех пакетов, включенных в каждый вариант DVD. Он слишком велик, чтобы привести его в нашем обзоре целиком. Достаточно сказать, что здесь есть все ваши любимые и часто используемые инструменты плюс несколько альтернатив к ним.

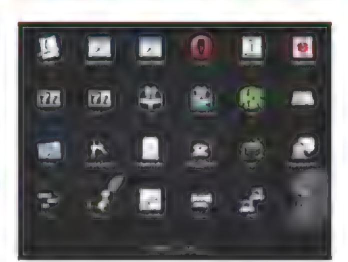
Ваши потребности в мультимедиа удовлетворят *VLC*, *Amarok*, *Xine* и другие. Интернет-приложения (web-браузеры, почтовые клиенты, мессенджеры и клиенты IRC и т. д.) и игры (сгруппированные в Аркады, Логические, Настольные и т. д.) численно превосходят практически все остальные типы приложений на DVD. Энтузиасты оценят также *GIMP*, *Blender* и ещё несколько инструментов для реализации вашей креативности.

В отличие от прошлых выпусков, Gentoo 20140826 не очень озаботился ощущениями пользователя. Когда мы запускали Gnome, он каждый раз падал или утомительно тормозил, особенно по сравнению с KDE, который выглядит наиболее отточенным среди включённых окружений.

Последний DVD, в отличие от предыдущего, не имеет инструментов для диагностики или восстановления системы. Мы не нашли даже *GParted* для просмотра локальных дисков и разделов. Вдобавок проект решил ограничить предлагаемую функциональность и теперь ставит своей задачей лишь показать умение приложений Linux. Если у вас есть какие-либо свои причины употребить Live-окружение, свежий Gentoo DVD, вероятно, не для вас. LXF

Свойства навскидку





Режим с сохранением
Вы можете использовать режим с сохранением данных на разделе того же USB-носителя, с которого загружаетесь...

Богатый ассортимент
Опыт с Gnome оказался малоприятным, зато о числе приложений вы можете судить по этому экранному снимку.



Gentoo 20140826

Разработчик: Gentoo Foundation и сообщество
Сайт: www.gentoo.org
Лицензия: Различные открытые лицензии

Функциональность	6/10
Производительность	7/10
Удобство в работе	8/10
Документация	9/10

» Находится в тени своего же выпуска от 2012 года, не предлагая ничего реально нового...

Рейтинг 7/10

Peach OSI 14.04

Хмурый вид **Маянка Шармы** можно понять: он нарвался на очередной дистрибутив для новичков с фруктовым привкусом.

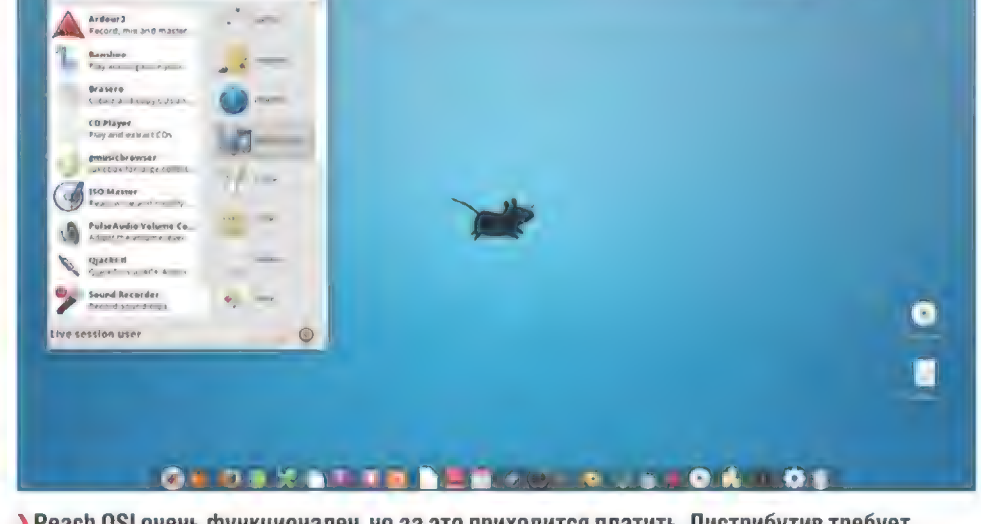
Вкратце

» Ещё один дистрибутив для новичков на базе Ubuntu. См. также: Zorin, PinguyOS, Elementary.

То, что дистрибутив назван в честь фрукта [peach — *англ.* «персик»], головой выдаёт его целевую аудиторию. Быстрый взгляд на сайт подтверждает, что Peach OSI 14.04 разработан для новичков в Linux. Вполне предсказуемо, там перечислены всё те же небезызвестные преимущества Linux перед проприетарными дистрибутивами: более безопасен, менее уязвим, более быстр на старом оборудовании и так далее. Так что мы загружали этот дистрибутив не без опаски. Однако уже через пять минут Peach завоевал наши сердца и жёсткие диски.

Peach OSI 14.04 основан на последнем Xubuntu 14.04 LTS, благодаря чему работоспособен на относительно старых компьютерах. Модифицированный рабочий стол *Xfce* имеет нижнюю панель, наполненную ярлыками приложений, которая прямо трещит по швам, так как ярлыков более 20 — от браузера *Firefox* до хранителя паролей *KeePassX* и менеджера фотографий *digiKam*.

Взглянув на меню, вы заметите, что дистрибутив с ходу включает чуть ли не каждую популярную программу для Linux, о которой вам приходилось слышать. Но разработчики не просто упаковали в комплект все приложения, до которых у них дотянулись руки. На самом деле, они весьма дотошно подошли к выбору и обосновали включение нескольких приложений для, казалось бы, одной и той же задачи. Например, по части Интернета, кроме браузера *Firefox*, прилагается и *Midori*, ибо последний идеален для создания web-приложений.



› Peach OSI очень функционален, но за это приходится платить. Дистрибутив требует гигантские 14,4 ГБ на диске.

Неочевидная логика

Сайт провозглашает, что Peach OSI содержит почти 100 приложений. Тут и популярные программы, которые имеются и в других дистрибутивах, такие как *LibreOffice*, *Thunderbird* и *Ubuntu Tweak Tool*, и менее привычные, но столь же полезные приложения вроде *Entangle* для связной фотосъёмки [tethered shooting], *Blender* для 3D-моделирования, *Ardour3* для работы со звуком, *wxBanker* для учёта финансов, а также *WINE* вместе с интерфейсами *Winetricks* и *PlayOnLinux* — для установки программ и игр с Windows.

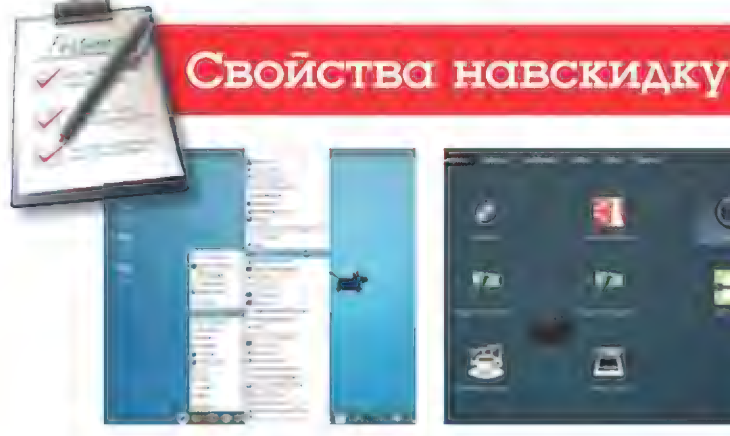
Некоторые из приложений для *Xfce* неродные и зависят от библиотек KDE или Gnome. Спасибо разработчикам за объединение этих библиотек без ущерба для удобства. Дистрибутив выглядит гармонично и функционирует исправно даже после нескольких часов работы.

Благодаря набору приложений, Peach будет практичным сразу из коробки для множества пользователей Linux. Имеется даже по несколько способов добраться до приложения. Дистрибутив содержит четыре разные «запускалки [launcher]» для разных пользователей или ситуаций. Есть здесь стандартное меню *Xfce*, в котором приложения разбиты по категориям. Есть и альтернативное меню *Xfce — Whisker*, с быстрым доступом к часто используемым программам. Также есть *Slingshot* в духе Mac OS X, позаимствованное из дистрибутива Elementary. И, наконец, старожилы Linux могут открывать список программ щелчком правой кнопкой мыши на рабочем столе.

Итак, ветераны-линуксоиды не будут разочарованы. Однако это не значит, что создатели Peach пренебрегают интересами своей основной аудитории. Вся документация на сайте дистрибутива нацелена на неопытных пользователей Linux и помогает им освоиться с незнакомыми понятиями вроде «сумма MD5», «репозиторий» и т. п. и разобраться в техническом жаргоне. Кроме того, на рабочем столе в режиме Live находится файл readme, который поможет пользователю-новичку пройти через установку обновлений и замену загрузчика.

Для первого релиза, Peach OSI 14.04 весьма неплох. Он стабильный, отзывчивый и очень быстрый, несмотря на насыщенность приложениями. Если хотите дать кому-то распробовать Linux на вкус, предложите ему Peach OSI. LXF

Свойства навскидку



Набит программами
Дистрибутив идёт с множеством предустановленных приложений из KDE, Gnome и *Xfce*.



Всем и каждому
Чтобы быть полезным в различных случаях, дистрибутив включает четыре запускалки приложений.



Peach OSI 14.04

Разработчик: Джеймс Карпентер [James Carpenter]
Сайт: www.peachosi.com
Лицензия: GNU GPL

Функциональность	8/10
Производительность	8/10
Удобство в работе	8/10
Документация	8/10

» Домашних пользователей Linux, не брезгующих дистрибутивами на базе Ubuntu, Peach заинтересует.



Рейтинг 8/10

Gnome 3.14

Масса улучшений и новых функций — признак того, что проект хотел подлизаться к **Шашанку Шарме**. Миссия выполнена!

Вкратце

» Один из популярнейших рабочих столов, в последние годы получающий больше пинков, чем букетиков. Сомнительный переход на Gnome Shell привёл к громкой критике. Но благодаря возвращению Debian на Gnome, а также прошлым стабильным релизам, всё может измениться. См. также: Cinnamon, Mate, KDE.

До появления сенсорных технологий рабочий стол состоял из обоев, иконок и панели задач. Со временем многие элементы менялись или добавлялись, но основные их свойства оставались теми же. Затем случился большой толчок, связанный с сенсорными дисплеями, и Gnome решил перейти от традиционного рабочего стола ко Gnome Shell. Столь резкая перемена заставила многих уйти на другие среды и даже породила два ответвления — Mint и Cinnamon. Однако, несмотря на критику, проект продолжает инновации, и Gnome 3.14 может похвастаться здравым сочетанием новых функций и массы улучшений, на радость и старым, и новым пользователям.

Gnome поддерживает множественные касания: интерфейсом рабочего стола и приложений, открытием меню и т. д. можно управлять жестами нескольких пальцев. Увы, жест для открытия Обзора [Activities Overview] не заработал на нашем устройстве так, как было обещано, но всё остальное работало безупречно. В программах вроде *Evince* и *Eye of Gnome* можно приближать, удалять и поворачивать документы и изображения. В будущих релизах жесты станут поддерживать ещё больше приложений. Разработчики могут использовать *GtkGestures*, чтобы добавить в готовые приложения на *GTK+* большинство стандартных многопальцевых жестов: перелистывание, перетаскивание, растягивание, поворот, касание.

Данный релиз сконцентрировался на облагораживании рабочего стола с целью улучшения функциональности. Так,



› Gnome 3.14 работает по проверенному сценарию для релизов открытых проектов, горстями принося новые возможности и улучшения, чтоб угодить авторам обзоров.

переделанное приложение Weather [Погода] использует новый фреймворк Gnome для геолокации и показа прогноза погоды. В режиме «Обзор» добавлены новые анимации; также появились новые переходы для открытия, минимизации и закрытия окон. Значительно улучшена визуальная тема, включая вид прогресс-индикаторов, диалоговых окон, управляемых мышью полей [spinner], меню и т. д.

В режим обзора были добавлены *Калькулятор* и *Часы*, так что вы можете производить вычисления или узнавать время в любом городе через строку поиска, не запуская приложений для этого.

Новое и улучшенное

Нечто новенькое появилось у большинства приложений. *Music* теперь позволяет создавать свои плей-листы и искать не только в локальной коллекции, но и по онлайн-источникам наподобие Jamendo. В *Photos* появилась поддержка Google — отныне вы можете добавить учётную запись Google и просматривать фотографии, загруженные посредством Picasa, Google+ или устройств на Android. Также поддерживаются Flickr и Facebook, но для этого надо сначала добавить учётные записи в системные настройки. А в *Software*, графическом инструменте Gnome для управления приложениями, обновлён внешний вид страницы, и для большего числа приложений стали доступны скриншоты, описания и рейтинги. Стало можно устанавливать здесь же расширения к программам.

Возможность сделать мгновенный снимок виртуальной машины, чтобы позднее иметь возможность вернуться к её точному состоянию, является одной из наиболее важных возможностей любой системы виртуализации. Теперь *Boxes*, менеджер виртуальных машин в Gnome, позволяет такие снимки делать, а также одновременно запускать несколько виртуальных машин в разных окнах.

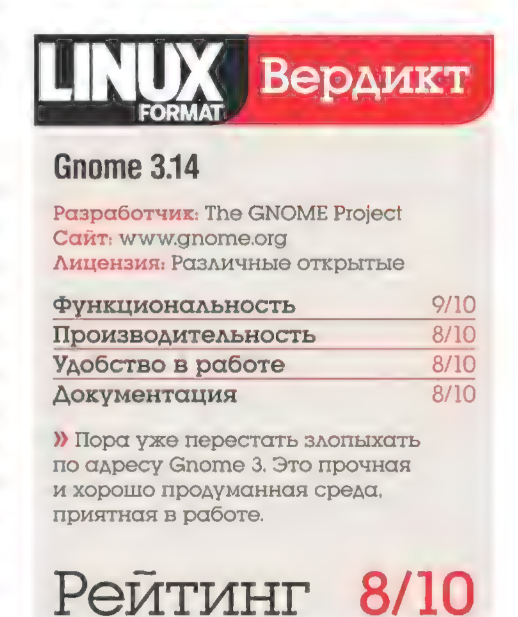
Что касается безопасности, в 3.14 появилась выдача доступа через сеть, то есть Gnome запоминает, в какой сети вы включали сервисы вроде WebDAV или VNC, чтобы случайно не запустить их, подключившись к публичному Wi-Fi.

Не считая мелких помарок, Gnome 3.14 оказался мощным, отлично сделанным релизом. Проект продолжает сглаживать шероховатости Gnome 3, предоставляя полезные функции и улучшая ощущения от использования. LXF

Свойства навскидку

Планы поездок
На *Maps* [Карты] теперь прокладываются автомобильные, велосипедные и пешие маршруты, с быстрыми подсказками.

Пальцы веером
В этом релизе добавилась поддержка нескольких жестов, чтобы разумно трудоустроить всю вашу пятерню.



LINUX FORMAT Вердикт

Gnome 3.14

Разработчик: The GNOME Project
Сайт: www.gnome.org
Лицензия: Различные открытые

Функциональность	9/10
Производительность	8/10
Удобство в работе	8/10
Документация	8/10

» Пора уже перестать злопыхать по адресу Gnome 3. Это прочная и хорошо продуманная среда, приятная в работе.

Рейтинг 8/10

Tonido для Raspberry Pi

Как матёрый пользователь *ownCloud*, **Маянк Шарма** интересуется, есть ли в этой области место проприетарному ПО.

Вкратце

» Сервер обмена файлами для установки на Raspberry Pi. См. также: *ownCloud*, *Seafile* и т.п.

В прошлом мы уже похваливали Tonido для Linux, а теперь этот бесплатный сервер для обмена файлами стал доступен и для Raspberry Pi. Если кто не слышал, Tonido выдаёт доступ к файлам и медиа с удалённого компьютера. Он также предустанавливается на «розеточных» мини-компьютерах [plug computer] с портами Ethernet и USB.

Простота Tonido сохраняется и в версии для Pi. Чтобы приступить к работе, достаточно скачать архив на Pi, распаковать его и запустить сервер. Затем можно настроить Tonido через удалённый браузер. По сути, нужно всего лишь создать уникальный TonidoID, идентифицирующий вашу установку Tonido на Raspberry Pi в сети. Этот ID используется промежуточным сервером Tonido, который позволит видеть Pi со всеми его общими файлами и папками откуда угодно в Сети без дополнительных настроек: никакой возни с брандмауэрами или динамическими DNS, просто выбираете уникальный TonidoID, после чего шустрый мастер настройки предложит вам выбрать папки для удалённого доступа. По умолчанию Tonido даёт доступ ко всем папкам, но вы, возможно, захотите это изменить, назначив папки вручную. Затем вам предложат указать, какие папки будут содержать Музыку, Фото и Видео. Эти настройки можно менять, через web-интерфейс Tonido.

Одна из самых восхваляемых функций Tonido — потоковое воспроизведение медиа-файлов. Если вы указали папки для медиа, Tonido просканирует их и добавит всё, что обнаружил, в соответствующие виртуальные библиотеки для быстрого доступа. Он также содержит встроенные медиа — и охотно воспроизвёл MP3 и MP4, но споткнулся на OGG и AVI. Разработчики предлагают установить *ffmpeg* для более полной поддержки мультимедиа.



› **Web-интерфейс Tonido очень интуитивен и включает собственный медиа-плейер.**

Делитесь как следует

Создать общую папку очень легко. Достаточно во встроенном менеджере файлов указать локальную папку на Raspberry Pi, которая может также находиться и на примонтированном USB-диске, и дать ей имя. Доступны также адекватные разрешения для публикации папки. Например, можно установить дату истечения срока публикации, дать доступ к папке всем, кто знает её URL, или оставить доступ только избранным лицам.

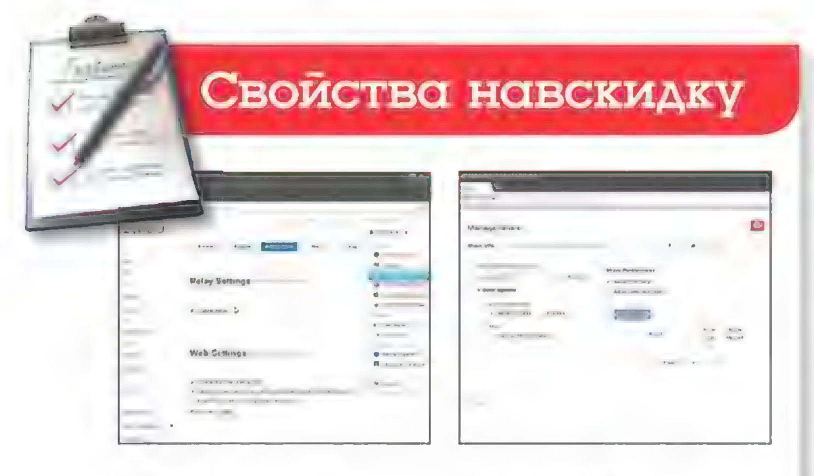
Выдав доступ только определённым пользователям, вы можете контролировать их право просматривать и загружать файлы индивидуально, а также ограничивать объём данных, который им разрешается загружать. Кроме того, Tonido хранит историю по опубликованным файлам. Единственное ограничение в бесплатной версии — на число добавляемых общих папок и гостей: не более пяти.

Чтобы загружать файлы, можно пользоваться встроенным файловым менеджером или просто перетаскивать их в окно браузера — но папки можно перетащить только в *Chrome*. Однако прогресс-индикаторы для наблюдения за ходом загрузки отсутствуют. Нет никаких признаков, позволяющих понять, загружается ли выбранный файл. Естественно, наибольшее неудобство это доставляет при загрузке больших файлов.

Для синхронизации потребуется установить клиент. У Tonido есть клиенты для Windows, Linux и Mac. Но клиент для Linux доступен только в виде файла **.deb** и только для 32-битных систем. Установив клиент, вы сможете синхронизировать файлы с Raspberry Pi без каких-либо проблем. Настроить путь к папке для синхронизации можно через web-интерфейс.

В конечном счёте, большинству пользователей Tonido поможет взлететь в облака. Впрочем, его удобство начинает таять по мере того, как вы становитесь более опытным и взыскательным... LXF

Свойства навскидку



Доступ отовсюду
Сервер TonidoID позволяет подключаться к вашему Raspberry Pi с любого компьютера с Интернетом.

Общие папки
Разрешайте доступ к данным, с легкостью создавая приватные и публичные папки.



Tonido для Raspberry Pi

Разработчик: CodeLathe LLC
Сайт: www.tonido.com
Лицензия: Проприетарная

Функциональность	6/10
Производительность	6/10
Удобство в работе	8/10
Документация	8/10

» Файловый сервер, простой в установке и настройке. Идеален для новичков, но плохо адаптируется к растущим нуждам.

Рейтинг 7/10

WordPress 4.0

Похоже, *WordPress* лихорадочно совершенствуется. **Шашанк Шарма** присмотрелся повнимательнее — в надежде, что и сам похорошеет...

Вкратце

» Популярная CMS, подходящая почти для всех видов сайтов благодаря обширной коллекции плагинов. См. также: *Drupal*, *Concrete5*...

Универсальность и приспособляемость сделали *WordPress* одной из самых востребованных систем управления контентом. Благодаря дизайну и простоте установки, *WordPress* привлекает всех пользователей, независимо от их навыков. А широкий выбор плагинов означает, что его можно подстроить под нужды почти любого сайта.

Авторы обзоров — порода разборчивая. Мы любим релизы, где появляется много функций, и воротим нос от тех, что добавляют мало новинок. Но в случае с инструментами вроде *WordPress* стоит смотреть не на отдельные изменения, а на то, как они влияют на пользовательский опыт от проекта. Это особенно верно для *WordPress 4.0*, названного Benny в честь джазового музыканта Бенни Гудмена [Benny Goodman].

В экосистеме открытого кода пользователям часто приходится самим делать выбор. Естественный отбор решает проблему, но только до некого предела. В основном, популярность проекта и его успех зависят от мер, которые принимаются для того, чтобы дать пользователям то, что они хотят, и делать так всегда. Именно этим *WordPress* отличался долгие годы, и последний выпуск не стал исключением.

Интернационализация всегда была сильной стороной проекта: около 30% сайтов на *WordPress* имеют неанглоязычный контент. В версии 4.0 *WordPress* надеется привлечь ещё больше пользователей со всего мира, предложив даже установку на любом языке. Помимо этого, поскольку у *WordPress* есть набор плагинов для борьбы со спамом, в 4.0 появилась возможность помечать как спам даже уже удалённые комментарии.



› *WordPress* вновь впечатлил стабильным релизом, где достаточно возможностей, чтобы понравиться прежним пользователям и впечатлить новых.

Уконтентовано

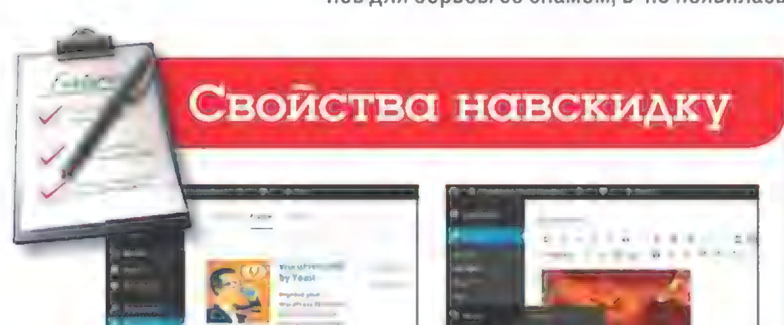
В релизе сделан упор на улучшение предпросмотра и редактирования данных. В прошлых версиях пользователю приходилось прокручивать страницу вверх, чтобы достать до меню управления редактированием. Теперь эта панель прикреплена к верхней части области редактирования, и работать стало комфортнее. Ещё одна новая функция — режим сетки [grid view], позволяющий быстро проглядеть все загруженные медиа-файлы. Можно также упорядочить список по типу, и даже отредактировать изображение, выбрав его эскиз и щелкнув на пункте Редактировать [Edit Image] в панели справа. При редактировании *WordPress 4.0* показывает все применяемые изменения «на лету». К сожалению, хотя режим сетки и позволяет выбрать сразу несколько файлов для вставки в пост или на страницу, возможности применить к ним одну и ту же операцию редактирования не предусмотрено.

При работе со встраиваемым контентом, будь то изображение или видео, *WordPress 4.0* теперь позволяет видеть их прямо в визуальном редакторе: можно вставить в область редактирования ссылку на картинку с Flickr, видео с YouTube или твит, и *WordPress* покажет её в редакторе, избавляя вас от необходимости нажимать кнопку предпросмотра, чтобы узнать, как встраиваемый контент смотрится вместе с остальным содержимым поста.

Плагины — это главная приманка пользователей *WordPress*. В последнем релизе доступные плагины отображаются в виде сетки. Карточка каждого плагина предоставляет кучу сведений о том, для чего он нужен, о его разработчике, совместимости с установленной версией *WordPress*, рейтинге и прочем. Кроме всего этого, релиз, как обычно, включает много исправлений ошибок и других изменений, призванных улучшить впечатление от работы.

Несмотря на устойчивость своего доминирования на рынке CMS, *WordPress* продолжает улучшаться с каждым выпуском, и 4.0 продолжает эту славную традицию. Мы полюбили Benny за то, что это завершённый, стабильный релиз. Вероятно, и вы полюбите! LXF

Свойства навскидку



Сетка плагинов
В сетке можно запретить показ плагинов, находящихся в стадии бета-тестирования.

Встраиваемые медиа
При вставке мультимедиа можно даже просмотреть видео или аудио прямо из визуального редактора.

LINUX FORMAT Вердикт

WordPress 4.0

Разработчик: WordPress Foundation
Сайт: www.wordpress.org
Лицензия: GPLv2

Функциональность	8/10
Производительность	9/10
Удобство в работе	9/10
Документация	9/10

» Не самый богатый функциональностью релиз, но WordPress 4.0 — безусловно нечто большее, чем сумма его частей.

Рейтинг 9/10

Tox

В извечном стремлении хранить тайну переписки **Маянк Шарма** тестирует клиент для шифрованных мгновенных сообщений, пришедший победить *Skype*.

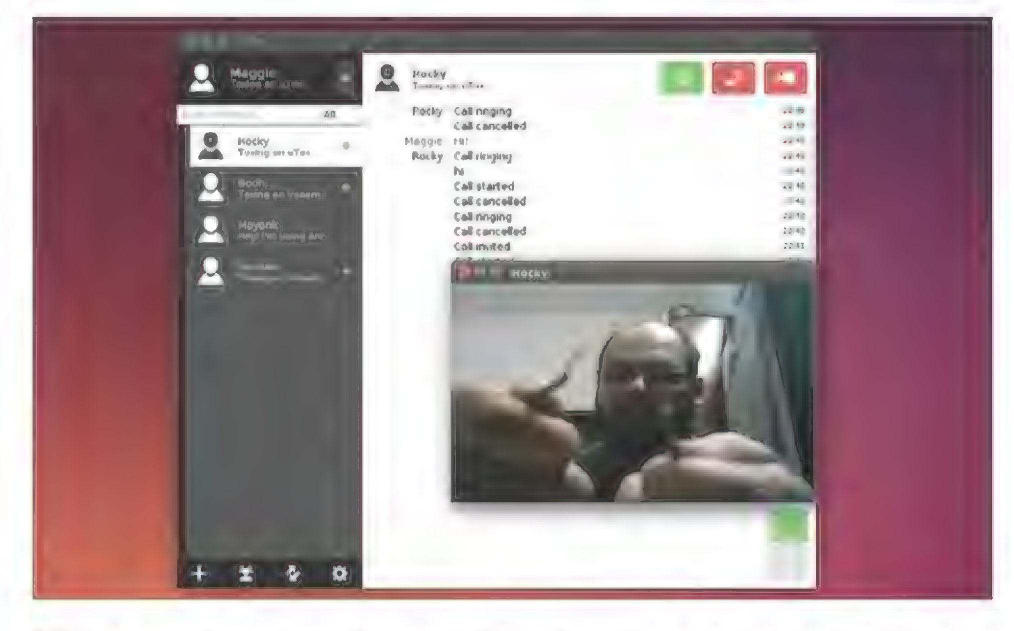
Вкратце

» Платформа для мгновенных сообщений, фокусирующаяся на приватности. См. также: Cryptocat, BitTorrent Bleep.

Мессенджер *Tox* — итог оживлённой дискуссии на 4chan о необходимости открытой, безопасной, децентрализованной замены *Skype*. *Tox* соответствует по всем этим пунктам. Для *Tox* важны две области технологий: шифрование и P2P. Клиент применяет ту же технологию, с помощью которой BitTorrent устанавливает прямые соединения между пользователями, без центрального сервера, который может оказаться взломан или отключен. Клиент использует идентификаторы Tox ID — это публичные ключи участников, а не учетные записи, что обеспечивает бóльшую анонимность. Более того, все чаты шифруются с помощью библиотеки *NaCl*.

Поскольку здесь нет центрального сервера, пользователи могут просто включить свои клиенты и добавлять друзей без регистрации и настройки на каком-либо сервисе. Каждый пользователь сети *Tox* представлен набором байтов — своим Tox ID. *Tox* использует распределённую хэш-таблицу [distributed hash table, DHT] в стиле торрентов, по которой участники ищут IP-адреса друг друга, используя Tox ID. Получив IP-адрес, участники создают защищённое соединение друг с другом.

На сайте проекта предлагаются клиенты для популярных настольных платформ: Linux, Windows и OS X, а также для мобильных платформ Android и iOS. Проект имеет репозитории APT и RPM, и в большинстве популярных дистрибутивов *Tox* установится без особой суеты. Добавив репозиторий, вы можете установить один из поддерживаемых клиентов: *µTox*, *qTox*, *Toxic* или *Venom*. Все они на разных стадиях разработки, но в наших тестах все настольные клиенты оказались производительнее и функциональнее, чем мобильные.



› Кстати, можно использовать *Tox* вместе с *Pidgin* через плагин *tox-prpl*: http://tox.dhs.org.

Удобство в работе у всех клиентов примерно одинаковое. Вы запускаете клиент, придумываете ник, после чего генерируется Tox ID, который можно передать друзьям. Когда друзья добавят вас, вы получите уведомление; его нужно принять, чтобы установить соединение. Если друг недалеко и он использует мобильный клиент, он может добавить вас, отсканировав QR-код, содержащий Tox ID.

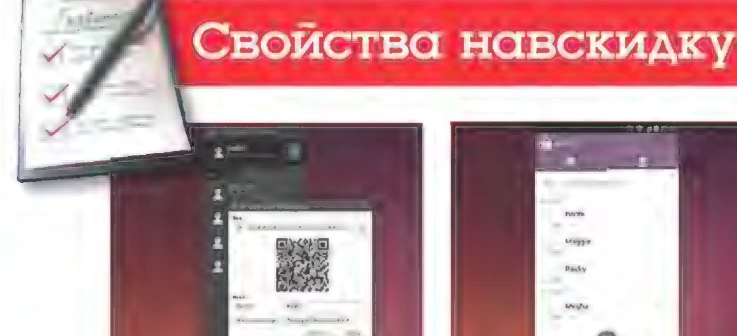
Детоксикация общения

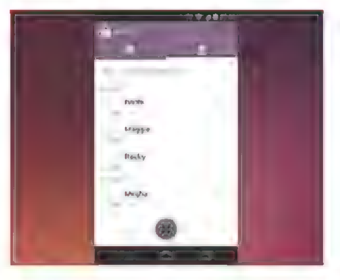
Мы протестировали *µTox* и *Venom* на настольном компьютере, а заодно и *Antox* на Android, и убедились, что разобраться в них очень легко. Текстовые сообщения работали на всех поддерживаемых платформах, так что пользователю *Venom* на Ubuntu можно отправлять сообщения пользователю *µTox* на Windows или *Antox* на Android. Пользователи всех трёх клиентов на настольных ПК могут совершать аудио- и видеозвонки между собой, и в нашем тестовом окружении не возникало заметных задержек или искажений. Аналогично, все клиенты могут передавать файлы: мы посылали всё, от мелких картинок до больших файлов ISO, и клиенты передавали их друг другу без проблем.

Но есть пара вещей, над которыми всё ещё ведётся активная работа. Самое важное — это возможность совершать аудио- и видеозвонки с Android-клиента *Antox*, которая пока отсутствует. Помимо этого, сейчас в групповых чатах с несколькими участниками вы можете обмениваться только текстовыми сообщениями.

Обычно мы не делаем обзоры программ, не достигших стабильного релиза. Но *Tox* — особенный случай, по двум причинам. Во-первых, он очень удобен, как минимум на настольном ПК. Во-вторых, это не только клиент — по сути, это P2P-протокол для передачи шифрованных данных. Некоторые разработчики используют этот протокол для создания безопасных почтовых клиентов или синхронизации файлов. А чтобы сообществу было спокойнее, разработчики планируют заказать в одной из компаний, занимающихся безопасностью, аудит своего кода, когда дойдут до более стабильной стадии. LXF

Свойства навскидку





Безопасная связь
Заботясь о безопасности, Tox использует алгоритмы криптографической библиотеки NaCl.

Кросс-платформенный
Клиенты для настольных и мобильных платформ, включая Linux, Windows, OS X, FreeBSD, Android, iOS.



LINUX FORMAT Вердикт

Tox

Разработчик: The Tox Foundation
Сайт: tox.im
Лицензия: GNU GPL v3

Функциональность	7/10
Производительность	8/10
Удобство в работе	7/10
Документация	6/10

» Находится на ранней стадии разработки, но очень удобен и эффективен, хотя пока только на настольном ПК.



Рейтинг 7/10

Untangle NG 11

Обзаводитесь новой сетью? **Маянк Шарма** рассматривает дистрибутив-шлюз, который развертывает сеть этак по-домашнему.

Вкратце

» Урезанный дистрибутив Linux, приспособленный для шлюзов. См. также: Zentyal и ClearOS.

Свежий выпуск дистрибутива-брандмауэра Untangle Next Generation Firewall v11, на базе Debian, доступен в двух редакциях: Бесплатной [Free] и Полной [Complete]. Первая — устанавливаемый ISO-образ. В отличие от настольных дистрибутивов, Untangle придётся сперва установить, а потом уж использовать. Предполагается, что дистрибутив будет работать сервером шлюза; установщик Debian здесь модифицирован и претендует на весь диск целиком. Надо только указать диск, разрешив стереть его.

После установки компьютер перезагрузится, перейдя в web-интерфейс мастера настройки. Вам предложат ввести пароль администратора и указать и настроить две сетевые карты — одну для Интернета и одну для локальной сети. Завершив настройку, Untangle попросит создать учетную запись (бесплатную), чтобы сконфигурировать сервер.

Войдя в свой сервер шлюза Untangle, первым делом установите требуемые вам приложения. «Рекомендуемая» установка позволяет одним щелчком добыть более дюжины приложений и сервисов, таких как web-фильтр, блокировщики вирусов и спама, контроль над приложениями, страница входа в сеть, балансировщик WAN, брандмауэр и другие. Не входят сюда, например, блокировщик рекламы, защита от вторжений и web-кэш.



› Большинство приложений в Бесплатной редакции — на самом деле 14-дневные пробники.

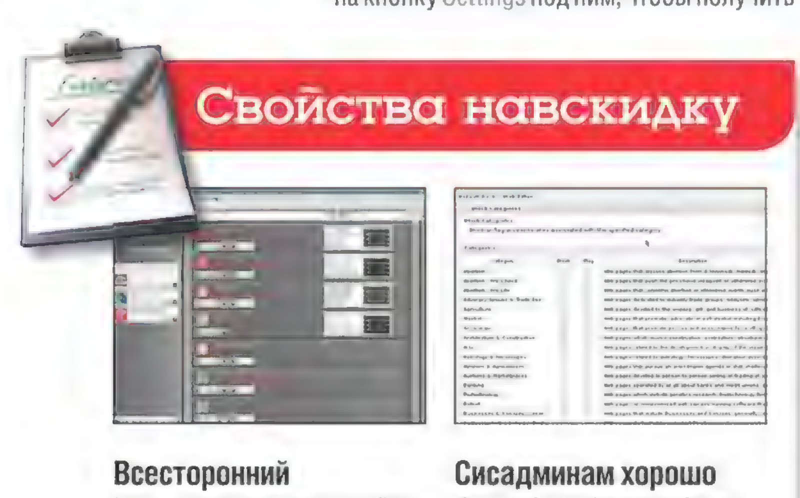
Распутать легко

Почти все приложения заранее настроены на автоматическую работу после установки. Настроить каждое можно, нажав на кнопку Settings под ним; чтобы получить справку, можно нажать на иконку помощи, которая ведёт на вики-страницу сайта Untangle, или же заглянуть на сайт, где есть демо-видео по более чем полудюжине самых полезных приложений.

Установив приложение, жмите на соответствующую кнопку Power, чтобы активировать службу. В этой консоли можно также анализировать трафик, проходящий через шлюз, и каждое приложение показывает всякую статистику по своему трафику: так, блокировщик рекламы сообщит количество заблокированных объявлений; антивирус — число сканированных страниц, а web-фильтр — заблокированных.

Однако большинство этих приложений и сервисов платные, а бесплатная редакция Untangle устанавливает их 14-дневные пробные версии. Вы можете купить лицензию на конкретное приложение; цены на них различаются и зависят от числа устройств в сети и периода подписки. Также можно приобрести Полную версию, со всеми платными приложениями, но цена снова варьируется в зависимости от этих двух факторов. Для примера, годовая подписка на Полную версию для 10 устройств будет стоить $540 (около €340). Этот пакет также включает дополнительные услуги, вроде индивидуального брендирования и поддержки.

Но стоит ли Untangle этаких денег? У него есть мощные соперники — Zentyal и ClearOS, и у обоих есть бесплатные версии от сообщества. Оба эти дистрибутива тоже уделяют много времени оттачиванию интерфейса приложений и поддерживаемых модулей, упрощая настройку и установку. Zentyal предлагает дополнительные платные сервисы, как и Untangle. Все трое предоставляют во многом схожую функциональность, но так как она реализована разными приложениями, какой-то продукт может справляться с определённой задачей лучше остальных. В конечном счёте, Untangle не предлагает ничего особенного, чего нет в нашем нынешнем любимчике Zentyal. LXF



Всесторонний
В комплект входят все обычные сетевые приложения, которые требуются от шлюза.

Сисадминам хорошо
Простой графический интерфейс через браузер очень облегчает установку и управление.



The Witcher 2

Сиквел игры балансирует на вершине успеха. А потом срывается. Ради возврата в игровую журналистику **Ричард Коббетт** нацепил свои доспехи +4.

Вкратце

» **Войдите вновь в образ Геральта из Ривии, великого Ведьмака, ещё более могучего, ещё более изощренного в своём деле истребления монстров, в этом кровавом, жутком, леденящем душу сиквеле.**

Спецификация

» ОС: Ubuntu 12.04+ или Steam OS
» CPU: 4-ядерный Intel
» ОЗУ: 4+ ГБ
» Графическая карта: GeForce 640+, поддержки Intel нет
» Драйвер: Nvidia 340.32, AMD fginx 14.14 rev. 2
» Поддержки драйверов MESA нет

Сиквел, устремившийся к Солнцу, в то время как его конкуренты продолжают пялиться на Луну. Это RPG AAA — игра с душой инди, переполненная захватывающими приключениями, которой можно насладиться, восхититься и, во многом, полюбить.

Во всём — от грубой технической магии движка до стенок палаток, подрагивающих на ветру, и поселян, бегущих в укрытие от дождя — эта игра делалась с пламенем страсти и стремлением к единственной цели: любой ценой стать лучшей RPG. До этого, признаться, не дотянули, но замах чертовски достойный — за 20–30 часов почти безоговорочно превосходных моментов, *The Witcher 2* поднимает планку практически во всём.

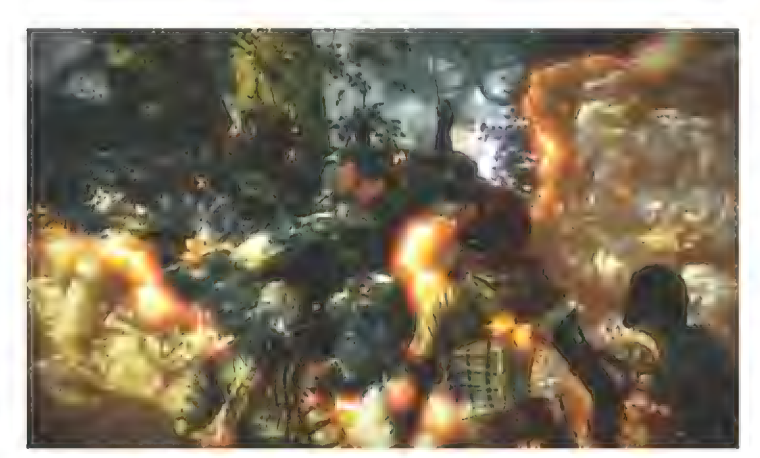


› **Монстров можно убить только серебряным мечом. А на людей он тоже годится?**

В этой сверхъестественной одиссее полным-полно ярких новаций и разветвлений главного сюжета, и в отличие от большинства RPG, они заходят гораздо дальше появления у вас баллов волшебной кармы или попытки неигрового персонажа вас облобызать.

В прологе, например, ваша цель — победить предателя, Ариана Ля Валетта [Aryan La Valette]. Убьёте ли вы его в поединке либо заставите сдаться, игра весело трюхает дальше. Масштабы последствий от ваших решений без малого смехотворные. Глава 1 предполагает два абсолютно разных финала, в зависимости от того, с кем вы работаете; оба драматичны и сделаны хорошо. Глава 2 — уже совершенно иной уровень, где вам предлагаются два ничем не похожих города, в зависимости от выбора, сделанного вами ранее.

Однако все эта детальность и амбиции имеют свою цену. Зачастую возникает ощущение, что то ли CD Projekt изо всех сил пытаются отступить от своей игры, то ли тестировщикам не дали взглянуть на неё свежим взглядом. Маркеры и описания квеста часто сбивают с толку, неверны или вообще отсутствуют — типовая ошибка по невнимательности того, кто заранее знает, что к чему. И в *The Witcher 2* слишком много отступлений и незримых препятствий. Но важно, что и повествование, и сюжет резко рвут с места в карьер — тут и столкновения огромных армий, и атаки драконов, и дерзкие побеги, а стартовое поселение изобилует страстями, интригами и интересными моральными дилеммами.



› **На горизонте появился *The Witcher 3*; будем надеяться, что он длиннее.**

В боевом арсенале всего понемножку. Как и прежде, фишка в том, что вы применяете стальной меч против людей и серебряный — против монстров, плюс несколько магических заклинаний, чтобы оглушить или сжечь врага и так или иначе склонить чашу весов в свою пользу. Это хорошо работает против одного-двух противников, но из-за долгих, непрерывных эффектов анимации и плохой системы планирования играть в группах может быть очень сложно. Что особенно проблематично на ранней стадии, когда Геральт [Geralt] ещё не набрал жизнеспособности, его заклинания слабы, и вам не блокировать более чем пару ударов за раз, тогда как атаки с тыла наносят 200% урон, и пара таких случайных встреч вас легко угробят.

Глава 1 была блистательной, увлекательной и душевной. Глава 2 — ещё лучше: эпична, драматична и удивительна. Казалось, что в Главе 3 игра вот-вот перейдёт те границы, где всё станет всерьёз, и вторая половина истории воплотится наяву... Этого не произошло. В Главе 3 нас ждала развязка, как будто *The Witcher 2* вдруг посмотрел на часы и сказал: «Ух ты, уже столько времени?»

Беда игры ясна: даже побив все рекорды своим пылом, страстью и стилем, она не смогла соответствовать своим же высоким стандартам. Учитывая все обстоятельства, простить это нетрудно. Забыть? Не так-то просто. Мы также должны вкратце упомянуть Linux-релиз, смешанный, на базе оболочки eON. Это даёт некоторые преимущества оборудованию Nvidia, однако большое обновление в августе и тонкие настройки драйвера AMD привнесли стабильности и производительности. LXF

LINUX FORMAT Вердикт

The Witcher 2

Разработчик: CD Project RED
Сайт: www.thewitcher.com
Цена: £14.99

Игровой процесс	8/10
Графика	9/10
Продолжительность	8/10
Стоимость	9/10

» Хотя мы и напоролись в конце на собственный серебряный меч, это одна из самых впечатляющих игр-ролевок.



Рейтинг 8/10

РАЗРАБОТЧИКАМ

Studio стабилизируется

Google окончательно перешел на новую среду разработки.

Праздник на улице разработчиков приложений для Android: Android Studio 1.0, интегрированная среда разработки для соответствующей платформы, сменила статус Release Candidate на стабильную версию. В основу продукта положен исходный код популярной открытой среды IntelliJ IDEA Community Edition от российской компании JetBrains.

Новых возможностей, в сравнении с кандидатом в релизы, немного. Однако практически все выявленные критические ошибки исправлены, и теперь Google официально считает Android Studio единым и единственным инструментом, который необходим при создании приложений для Android. Как известно, стандартной конфигурацией системы программных средств для разработки Android-приложений долгое время служил Eclipse с плагином ADT, теперь же его поддержка Google прекращена в связи с развитием более гибкой среды. Переход с Eclipse упростит предлагаемая Google справочная ссылка (http://developer.android.com/sdk/installing/migrate.html). Возможности Android Studio позволяют создавать программы не только для смартфонов или планшетов с разными разрешениями экранов, но и для телевизоров, «умных» часов, автомобилей и прочих устройств на базе Android.

Среди возможностей Android Studio — визуальный редактор для компоновки типовых элементов интерфейса; мастер создания собственных элементов оформления; встроенная интеграция с Github для загрузки типового кода; мониторинг использования памяти разрабатываемым приложением; унифицированная система сборки на основе Gradle; мгновенный доступ к облачным сервисам Google.



› User interface design — редактирование и просмотр по нескольким размерам экрана, языкам и даже версиям API.

ИНТЕРНЕТ ВЕЩЕЙ

Мы едем, едем, едем

Планируется интеграция Android в автомобиль.

Google планирует выпустить версию Android Auto для головных устройств автомобильных информационно-развлекательных систем, которая обеспечит нативный доступ к Интернету и мобильным приложениям без необходимости подключения смартфона.



› Android M — ваш автомобиль становится большим мобильным устройством.

Сегодня CarPlay от Apple и стандарты Open Source, такие как MirrorLink или GENIVI, отображают версию смартфона ОС на головном устройстве информационно-развлекательной системы автомобиля, позволяя водителю пользоваться такими приложениями, как Google Maps или iTunes. Анонсированная Google прошлым летом система Android Auto, первые автомобили с которой дебютируют уже в 2015 году, также позволяет управлять подключенным Android-смартфоном через экран головного устройства. Теперь же, по информации агенства Reuters, Google планирует встроить Android Auto (возможное название будущей системы — Android M) непосредственно в автомобиль.

Вице-президент исследовательской фирмы Gartner Inc. Тило Козловски [Thilo Koslowski] напомнил, что эта новость не является сюрпризом: идея прозвучала летом 2014 г. при создании Открытого автомобильного альянса [Open Automotive Alliance], в который вошли Google, Audi, General Motors, Honda, Hyundai и Nvidia. «Ваш автомобиль становится большим мобильным устройством. Я называю это Интернетом автомобилей, который является частью Интернета вещей».

Прямая интеграция Android M в автомобиль гарантирует, что водитель будет пользоваться услугами Google всякий раз, включая зажигание. Это позволило бы Google, используя данные от камеры, датчиков, указателя уровня топлива и средства связи с Интернетом, предоставлять водителю разноплановую и актуальную для него информацию. Однако, по мнению аналитиков, планы Google могут столкнуться с различными техническими и бизнес-проблемами, в том числе с необходимостью убедить производителей настолько плотно интегрировать свои услуги в выпускаемые ими автомобили.

Источник: android-developers.blogspot.ru; www.reuters.com

КОНКУРЕНЦИЯ

Почта оживляется

После долгой стагнации почтовики развили бурную деятельность.

Стремясь к расширению почтовой функциональности пакета *Office* для платформ не-Windows, Microsoft приобрела Acompli — американский стартап, реализующий почтовый клиент для iOS и Android.

Основанный в 2013 году, Acompli предлагает мобильный почтовый клиент для iPhone и Android, в котором в единый оптимизированный интерфейс объединены также контакты и календарь. По отзывам пользователей, особенно хорош *Acompli* при работе с почтовыми серверами Microsoft Exchange. В то же время, предложенная Microsoft базовая версия *Outlook* для iPhone встречена пользователями достаточно прохладно.



› Про *Acompli* говорят, что это *Outlook* для iPhone, который еще не успела сделать Microsoft.

Microsoft планирует объединить группу разработчиков Acompli с командой собственного клиента электронной почты *Outlook*. Условия сделки не разглашаются, однако ресурс recode.net сообщает о сумме, превышающей $200 млн. По мнению аналитиков, наиболее вероятна интеграция *Acompli* в состав пакета *Office 365*. При этом нынешним пользователям *Acompli* по-прежнему будут доступны все услуги приложения и доступ к учетным записям, заверил соучредитель и генеральный директор компании Acompli Хавьер Сольтеро [Javier Soltero].

После многолетней стагнации на внедрение новой функциональности электронная почта превращается в поле конкурентной битвы между основными поставщиками услуг корпоративного программного обеспечения, активно внедряющими современные технологии. Так, в ноябре IBM представила Verse — комплекс для социального взаимодействия, использующий встроенные инструменты аналитики для организации общения, поиска нужных контактов и информации, а также работы над проектами. Радикальную модернизацию своей почтовой службы также предпринимает Google.

Согласно данным Radcati Research Group, ежедневно в мире отправляется свыше 108 млрд электронных писем.

НОВАЯ ВЕРСИЯ

Навигатор — безвозмездно

Но за полную функциональность придется заплатить.

Компания NAVITEL обновила версию навигационной системы Навител Навигатор 9.4 для ряда платформ, в том числе для Android. В версию 9.4.0.28 добавлена лента новостей — теперь все новости компании о выпуске обновлений программы и карт будут сразу доступны на устройстве в разделе Мой Навител - Новости. Сервис Навител.Друзья обзавелся фоновым режимом: выбрав в настройках эту опцию, ваши друзья смогут наблюдать за вашими перемещениями, даже когда Навител Навигатор не запущен на устройстве. При регистрации в сервисе Навител. Друзья добавлены в параметры пол и электронный адрес. Ускорены адресный поиск на картах формата nm2 и прокладка маршрутов, снижено потребление памяти, улучшен алгоритм построения маршрутов с учетом информации о пробках.

Для устройств на платформе Android доступна фримиум-версия Навител Навигатор 9.3.0.187 с бесконечным бесплатным периодом использования. По окончании 7-дневного полнофункционального периода произойдёт переход в режим с ограничениями на прокладку маршрутов, голосовое сопровождение, 3D-режим карты, информацию о дорожных заторах (пробках), указатели движения по полосам. Чтобы получить полнофункциональную премиум-версию, надо купить лицензионный ключ.

В новой версии Навител Навигатор 9.3.0.187:

- » улучшено потребление памяти кэшем файлов;
- » снижено энергопотребление программы;
- » исправлены ошибки визуализации маршрута;
- » улучшена навигация в тоннелях;
- » исправлены ошибки адресного поиска при работе с атласом из нескольких карт;
- » внесены другие мелкие исправления, повышающие стабильность и надежность работы программы. LXF



› По окончании 7-дневного полнофункционального периода Навител Навигатор 9.3 переходит в режим работы с ограниченной функциональностью.

Источник: recode.net; navitel.ru

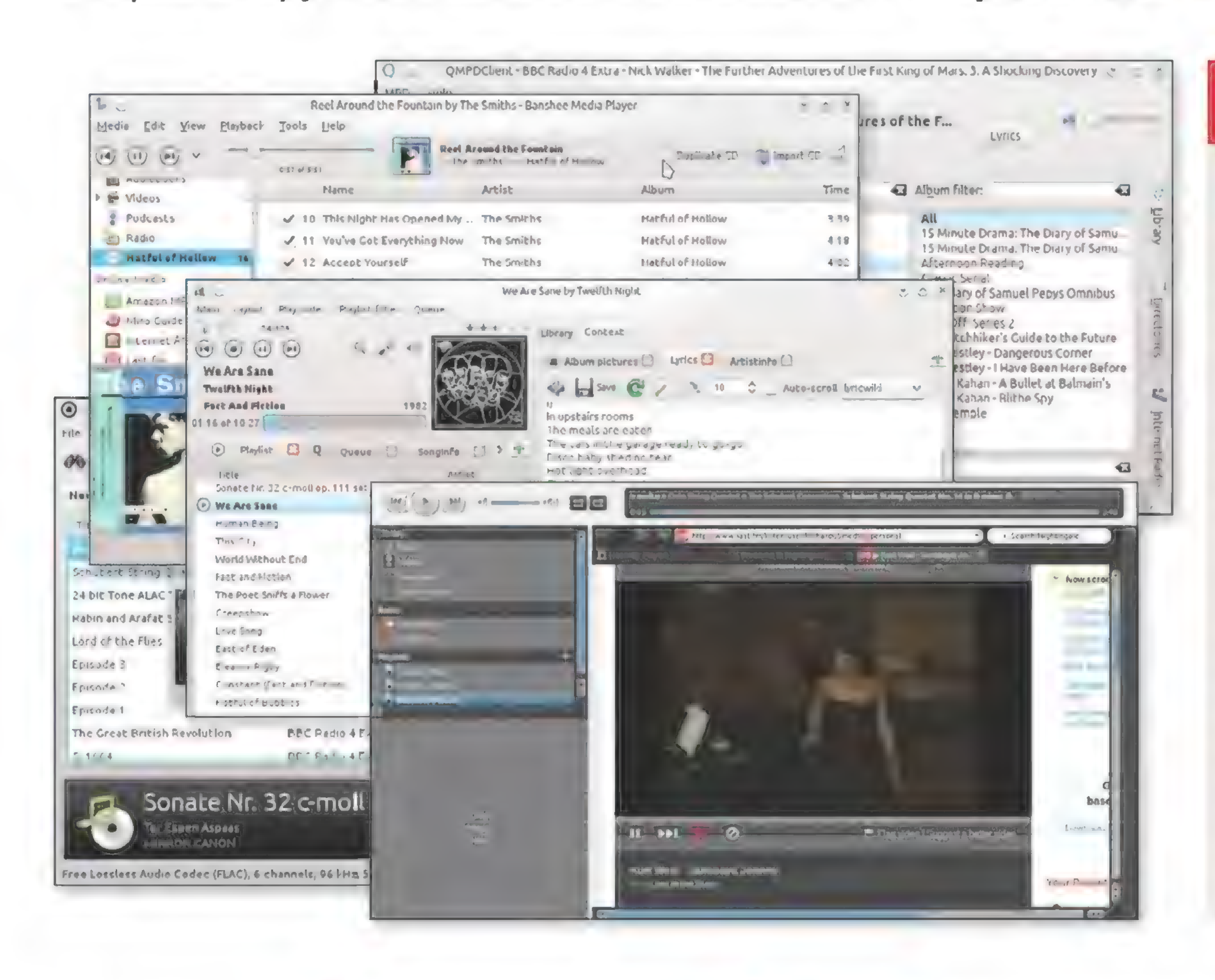
Сравнение



Каждый месяц мы сравниваем тонны программ — а вы можете отдыхать!

Музыкальные плейеры

Какие музыкальные плейеры справятся с большими фонотеками, сочетая это с хорошей функциональностью и мощью? **Ричард Смедли** обратился в слух...



Про наш тест...

В наших плейерах немало функций, но нам хотелось дать реалистичную оценку, поэтому мы тестировали на относительно скромном двуядерном ноутбуке с 2 ГБ ОЗУ, а чтобы имитировать среднестатистический целевой компьютер, мы также запустили их на первый Acer Aspire One Netbook. ОС была любезно предоставлена Ubuntu 14.04, Debian Testing (Jessie) и Fedora FC20. Иногда мы слушали воспроизведение в минимальном менеджере окон (обычно *XMonad*), а иногда в более реальной среде, KDE или Unity, при других приложениях, тоже потребляющих ресурсы. Использовались файлы с высоким разрешением, а также CD; FLAC без потерь; сжатые закачки (MP3 и AAC) и потоки. Файлы были на внешнем диске и принимали участие на всех машинах, заодно позволив проверить, как работают поиск и создание плейлистов вне домашней директории.

Наша подборка

- Audacious
- Banshee
- GMusic-Browser
- MPD/QMPDClient
- Nightingale

Даже если превыше всего вы цените качество звука на виниле, вам все равно не избежать скапливания множества цифровых файлов — музыки, аудиокниг, подкастов и прочих аудиопрелестей — доступных исключительно в виде того или иного цифрового формата.

Большинство дистрибутивов заботятся о поддержке кодеков для вас, и аудиовещание в Linux справляется с аудиофайлами высокого разрешения, например, 24-битными с частотой выборки 192 кГц, а значит, хорошее качество звука можно получить практически во всех программах. Исключение здесь — Direct Stream Digital (DSD), поскольку поддержка была добавлена в Advanced Linux Sound Architecture (ALSA) ядра всего год назад.

> «Здесь мы сравниваем самые богатые функциональностью аудиоплейеры...»

Опций для хранения и воспроизведения множество, со специальными компонентами Hi-Fi, конкурирующими с несколькими программными решениями для трех основных платформ ПК. Здесь мы сравниваем самые богатые функциональностью аудиоплейеры, способные справляться с огромными подборками музыки — десятками тысяч треков: за эти пределы ваш плейер по умолчанию вряд ли выйдет.

Есть масса решений для тех, кому нужен более минималистский аудиоплейер; но для типового применения, например, воспроизведения музыки с жесткого диска вашего ноутбука или нетбука, присоединенного к усилителю, мощь и гибкость этих пяти программ выводит их вперед.

Удобство в работе

Интуитивный — настолько затертое слово...

У *Banshee* очень гибкий интерфейс. Его многопанельный интерфейс размещает все именно там, где это легко достать курсором — кнопки управления и меню вверху, выбор медиа сбоку и три панели с информацией доступны по требованию, так что переключаться между его многочисленными сервисами легко. Неудивительно, что это удобное приложение быстро стало медиа-плейером по умолчанию для многих пользователей Ubuntu, и не только. Сообщалось о нестабильности более ранних версий *Banshee* на рабочих столах KDE, однако мы обнаружили, что это уже не так — хотя надо признать, что у нас был сбой при усиленной нагрузке на системные ресурсы; той же хворью страдает *Nightingale*.

У *Nightingale* понятный интерфейс по умолчанию, сосредоточенный на плей-листе и импорте, сортировке и воспроизведении плей-листа. Поиск текста по перечисленным провайдерам имел переменный успех, а обложки альбомов отображались не всегда; но с точки зрения чистого аудио, *Nightingale* можно было заставить делать все, что угодно, без проблем.

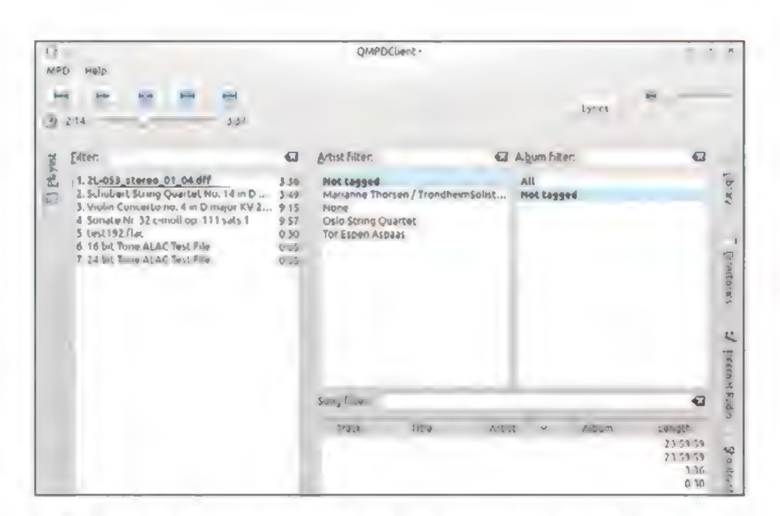
MPD (*Music Player Daemon*) — особенный случай, потому что доступ к нему вы получаете через клиент. Мы сосредоточились на *QMPD*, клиенте на базе *Qt*: при своей обманчиво простой внешности он предлагает неплохой выбор функций. Доступны также клиенты *GTK* и терминала, но для работы с большими коллекциями музыки важен сам *MPD*, а клиенты различаются по наборам функций, скорости и удобству.

У *QMPD* замечательно четкий многопанельный интерфейс с немногими меню, значками опций управления плей-листом и вкладками для библиотеки, плей-листов, интернет-радио и т. д., привязанными к вертикальной колонке справа. Тем не менее с настройкой *MPD* воспроизведение музыки *QMPDClient* приятно удивляет простотой, а программа оказалась очень быстрой и отзывчивой.

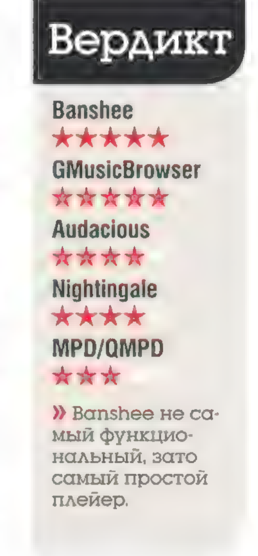
Несколько старомодный интерфейс *Audacious* был освежающе прям, и хотя его можно заменить на классический голубой неон в стиле *WinAmp*, мы предпочли работать с его темой *GTK* по умолчанию. Расширяемые меню, основные значки воспроизведения, плей-лист и цветная панель обложки и традиционная визуализация спектра создают простой в использовании плейер.

GMusicBrowser, хоть и имеет довольно насыщенный интерфейс, является еще одним плейером, сосредоточенным на плей-листе и индивидуальных треках. Мы обнаружили, что переход из библиотеки на вкладку контекста устраняет большую часть отвлекающих моментов, и те, кто любит ориентированные на контекст опции правой кнопки, почувствуют себя в этом плейере как дома.

GMusicBrowser в этом состязании делит лавры с *Banshee*...



› **Несмотря на скудный интерфейс и настройку файла config, *MPD* оказался единственным плейером, который справился с файлами DSD.**



Вердикт

Banshee ★★★★★
GMusicBrowser ★★★★★
Audacious ★★★★
Nightingale ★★★★
MPD/QMPD ★★★

» Banshee не самый функциональный, зато самый простой плейер.

Настройка

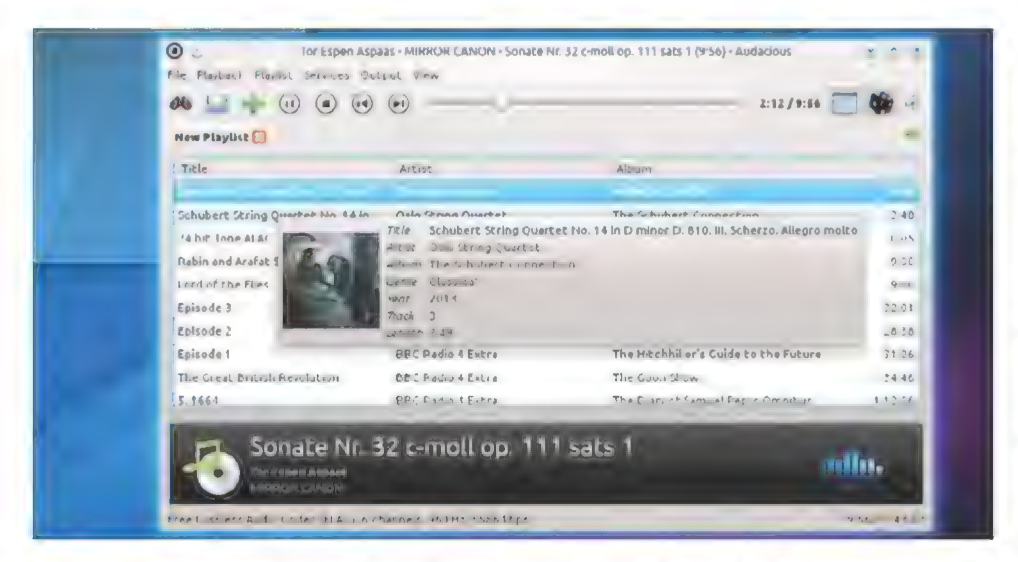
Поиск музыки и управление ею.

В *anshee* способен установить связь со многими сервисами, и в том числе предлагает поиск по MP3 с Amazon, плей-листам Last.fm, Miro Guide и DLC из Internet Archive. *Audacious* тоже отличился легкостью настройки плей-листов. Несложный интерфейс в очередной раз демонстрирует, как легко раскрыть возможности плейера, а список установленных плагинов впечатляет.

Неудивительно, что, будучи серверным приложением, *MPD* требует определенной настройки. Надо отредактировать **/etc/mpd.conf**, указав абсолютный путь к месту пребывания музыки, затем запустить **mpc update**. Имеется выбор клиентского ПО; *QMPDClient*, описанный здесь — разумный выбор для пользователей KDE...

GMusicBrowser примечателен простым интерфейсом по умолчанию и простой настройкой с разными опциями сделать его похожим на большинство других медиа-плейеров, или, подобно *Nightingale*, он использует среду GStreamer заодно и для обработки медиа; то есть поддержка кодеков простирается насколько душе угодно, и разрешает использование на вашем ПК несвободных кодеков (через подборку *gstreamer-plugins-ugly*, которую пользователи Ubuntu обнаружат в репозитории Multiverse).

Nightingale оказался менее интуитивным, чем *Banshee* и *Audacious*, но хорошо работал во время настройки и процесса поиска — по крайней мере один раз нам пришлось переключиться на пакет non-Unity и перезапуститься с настройками US-English.



› ***Audacious*, как и все плейеры, работает с аудиофайлами 192 кГц/24-бит HD.**

Вердикт

Audacious ★★★★★
Banshee ★★★★★
GMusicBrowser ★★★★★
MPD/QMPD ★★★
Nightingale ★★★

» Audacious воспроизводил даже файлы звукового чипа Commodore 64.

Функции и расширения

Вам нужен будильник? Не проблема.

Мы ожидаем от музыкальных плейеров намного большего. Отображение обложки и текстов воспринимаются как должное, так же, как и скробблинг Last.Fm и похожих сервисов.

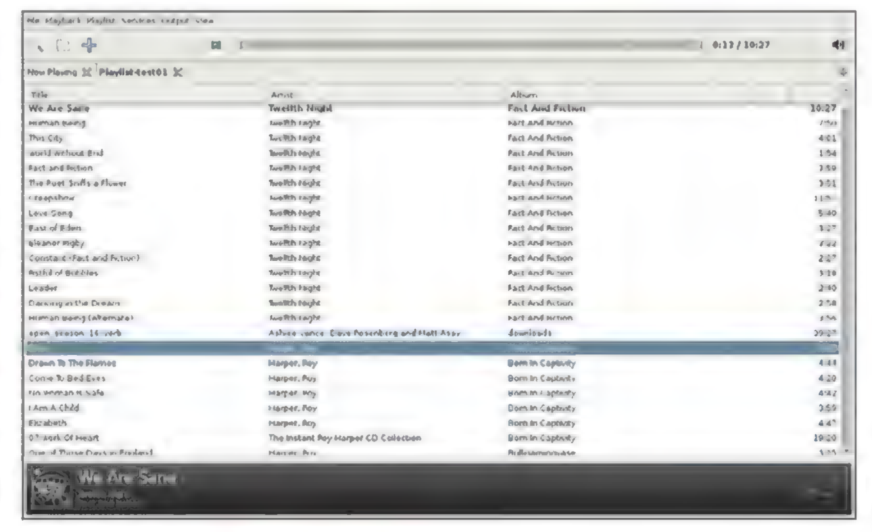
Попытки продемонстрировать свою красоту (по крайней мере, для программного обеспечения) ограничиваются чисто внешним: массой тем, гармонирующих с вашим любовно настроенным рабочим столом, особенно благодаря наличию тысяч созданных пользователями скинов.

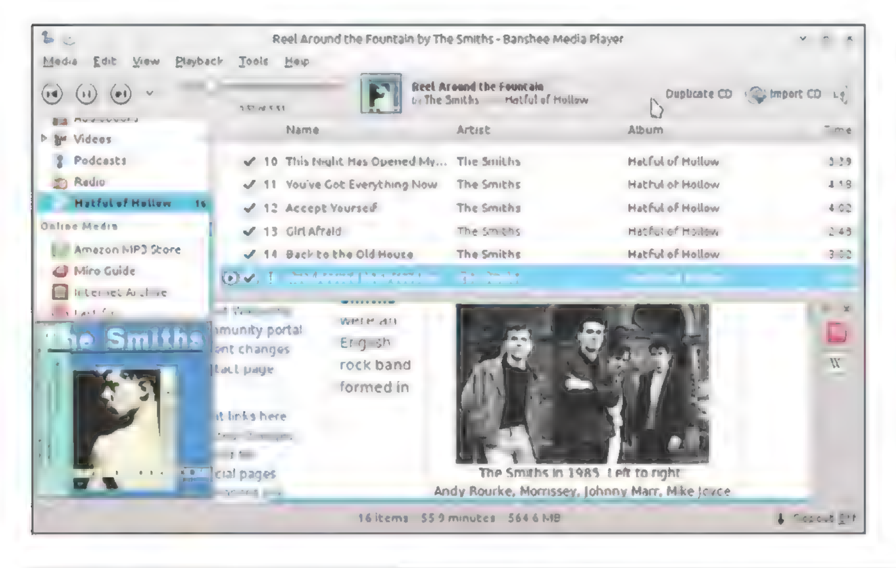
Наши пять плейеров определенно имеют все необходимое, чтобы управлять плей-листами и укротить ваш 250-ГБ диск с файлами MP3; но коль скоро свободные программы и программы с открытым кодом обладают потенциалом многофункциональности, почему бы не рассмотреть его? Здесь могут быть дополнительные функции, способные повлиять на ваш выбор, и они разнообразны: от поиска музыки с лицензией CC (Creative Commons) до воспроизведения файлов DSD, каждый плейер предлагает некую изюминку, благодаря которой он выделяется на фоне соперников...

Audacious 

Мы уже упоминали плагины *Audacious*, и за его скромным фасадом прячется плейер, готовый взяться за что угодно. Здесь есть удобные аудиоэффекты, типа стереофонической конверсии (заставляющей наушники звучать «реалистичнее») и эффектов LADSPA. При запуске открывается окно EQ, что где-то и чрезмерно, если у вас хорошо сбалансированы усилитель и колонки. Впечатляет поддержка кодеков, которая варьируется от Nintendo DS Sound Format до Apple Lossless (ALAC), и есть даже плагины для таких форматов, как M3U, PLS и XSPF.

Помимо множества непонятных кодеков и плагинов, *Audacious* позволяет изменять свой внешний вид скинами *Winamp2* (файлы WSZ). Настраиваемый пользователем цветовой баланс означает, что возможности безграничны, однако большой контроль означает большую ответственность, к огорчению всех наблюдателей.





Banshee 

К лучшей совместимости с переносными музыкальными плейерами *Banshee* добавляет умные плей-листы и утилиты для работы с большими коллекциями, при таких функциях, как пакетное восполнение недостающих метаданных. Используя Musicbrainz, *Banshee* автоматически находит отсутствующие и дополнительные метаданные для элементов библиотек, включая обложки. А если у вас есть мультимедиа-ключи, настроенные в Gnome, *Banshee* и их использует. Подобно *Audacious*, *Banshee* умеет нарезать CD. Среди достойных упоминания расширений — Karaoke; Stream Recorder для записи потоков Интернет-радио; и LCD, отображающее информацию о треках на ЖК-экране с помощью LCDproc, так что *Banshee* — отличный выбор для проекта встроенного музыкального плейера. Есть и добавочные элементы, например, ввод с опережающим поиском и отличная поддержка аудиокниг и подкастов; поэтому *Banshee* во многих дистрибутивах является плейером по умолчанию.

Работа с крупной фонотекой

Если вы накопили более 5000 музыкальных CD, то это для вас.

Наши пять плейеров были выбраны благодаря своему умению работать с большими коллекциями музыки — способными заполнить недавний диск, на что сетует iPod Classic. *GMusicBrowser* здесь выделяется своим гибким сканированием. Добавьте сюда функции настройки плей-листа и такие штрихи, как опция воспроизведения во взвешенном случайном порядке — и это отличный шанс насладиться своей коллекцией.

Audacious быстренько проиндексировал большой внешний диск и всё подряд и без разбору добавил в плей-лист, не отличая аудиофайлы — правда, удалить лишнее из плей-листа несложно. *Nightingale* тоже индексирует быстро, но съедает немало системных ресурсов, независимо от размера музыкальной коллекции, с которой работает, поэтому на слабом оборудовании он тормозил.

Banshee приотстал от других четырех плейеров, но если отставить в сторону странный сбой, он храбро продвигался вперед. Его простой интуитивный интерфейс извиняет прочие недостатки.

Ожидалось, что *MPD* справится с тестами хорошо; так и вышло. Однако он работает не как другие плейеры, которые создают плей-листы, сканируя директории в поисках медиа-файлов. *MPD* работает из одной директории с музыкой. Если вы храните звуковые файлы где попало, понадобится создать символические ссылки из приписанной к *MPD* директории, хотя он может читать с удаленных серверов через СМВ или NFS. Это рекомендуется при применении *MPD* на более скромном оборудовании, вроде Raspberry Pi.

Вердикт

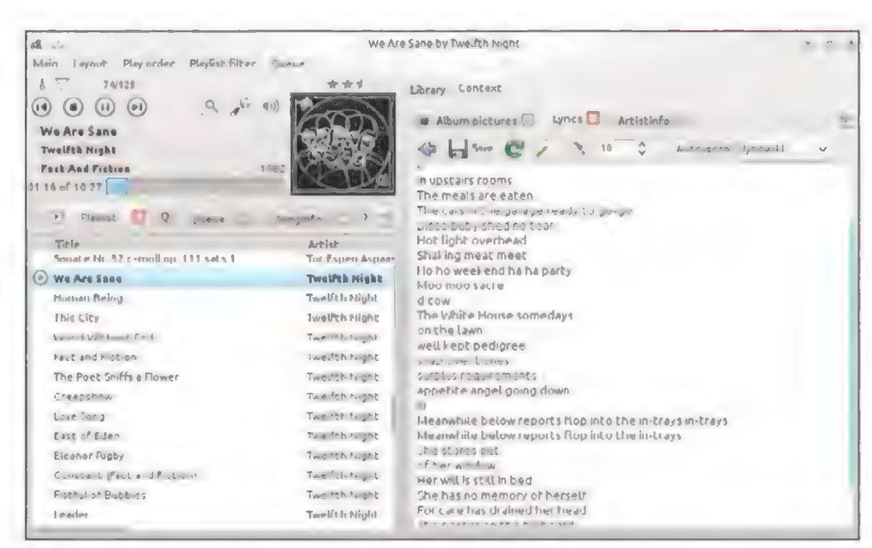
GMusicBrowser
★★★★★
MPD/QMPD
★★★★★
Audacious
★★★★
Nightingale
★★★★
Banshee
★★★

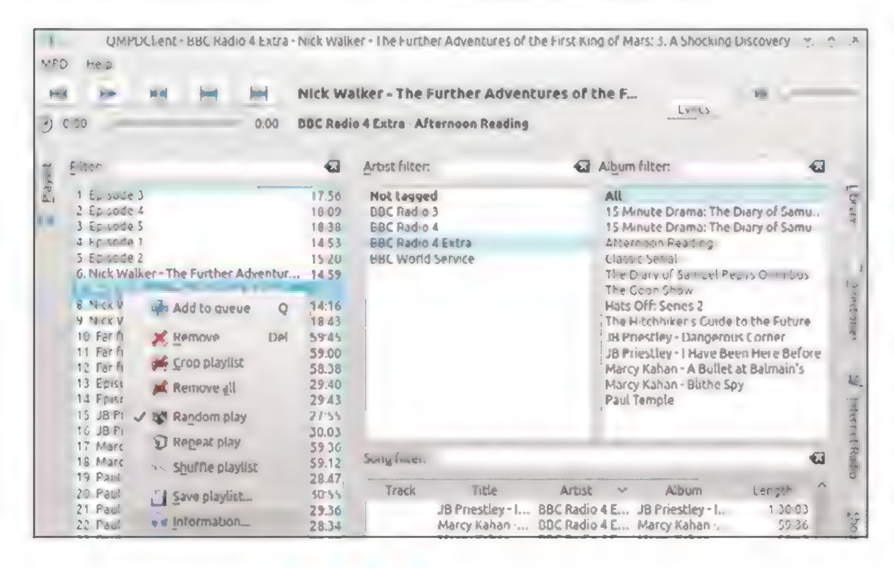
» Простота GMusicBrowser оттесняет назад настраиваемость MPD из командной строки.

GMusicBrowser 

GMusicBrowser позиционирует себя как «музыкальный автомат для больших коллекций MP3, Oggs, FLAC и MPC». Плей-лист — сердцевина всего, что он делает, и ничто не сравнится с ним по гибкости и мощи. Такие функции, как простые пакетные маркировка и переименование, а также индивидуально настраиваемые ярлыки, облегчают управление. С плейером поставляется тема Shimmer, улучшающая стандартную раскладку кнопок управления и их внешний вид, и в ней предусмотрен ряд интересных опций, например, мозаика из обложек альбомов.

Плагинов меньше, чем в *Audacious*, однако тексты песен и обложки они находят. Особо стоит обратить внимание на Albumrandom3, который воспроизводит весь альбом во взвешенном случайном порядке, и Sunshine3 — будильник, плавно переходящий в музыку на заре вашего дня (он также заставит треки постепенно затихать, когда вы укладываетесь спать).





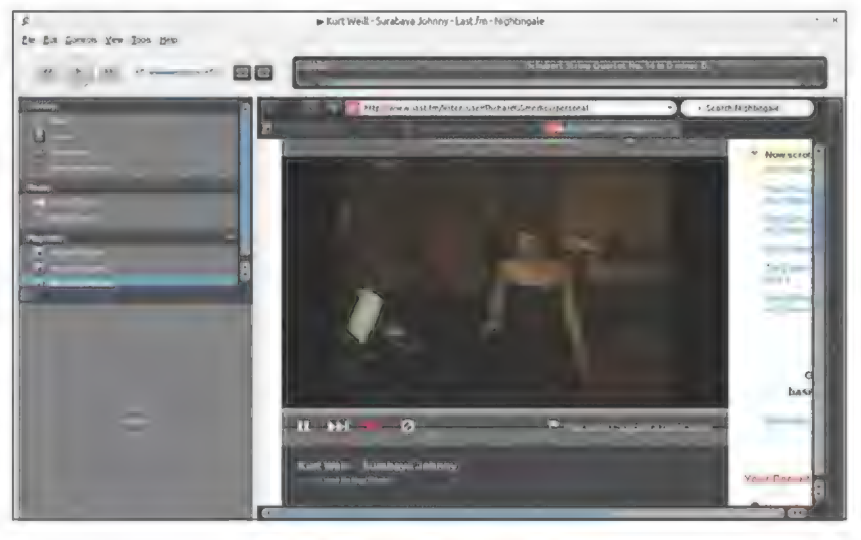
MPD/QMPDClient 

Сила *MPD* в том, что он работает с десятками тысяч треков, даже не вспотев, а клиенты вроде *QMPDClient* берут на себя тяжелую работу, обеспечивая его функциями. Функции EQ в клиенте нет, однако *QMPDClient* предлагает множество других, в том числе обложки, тексты песен, интернет-радио, плейлисты, треки Last.fm, уведомления в системном лотке (а также Freedesktop), скины и просмотр директории Shoutcast.

MPD был единственным из наших плейеров, который справился с файлами с высоким разрешением Direct Stream Digital (DSD). Это формат Super Audio CD, и он предлагает более «естественный» звук, чем кодирование PCM, применяемое в других дисках и закачках. Если вас устраивает CD или сжатое аудио, то и ладно; однако при быстром росте библиотеки треков DSD, уже доступных для приобретения и последующего скачивания, меломаны могут качнуться в сторону *MPD*.

Nightingale ★★★★

Nightingale подхватил исходный код *Songbird* и улетел с ним, но поддерживает совместимость и с плейером, и с его расширениями. Вы можете надставить интерфейс более важной информацией о треке (или аналоговыми часами, если вам так нравится); искать музыку с лицензией CC; находить битые, отсутствующие или повторяющиеся треки; добавлять отображение в виде сетки в стиле iTunes и даже находить гитарные табы. Секрет его расширяемости — в *Firefox*-подобной природе его расширений, поскольку *Nightingale* использует Mozilla XULRunner, что делает встроенные окна браузера естественной функцией, и вы найдете такие вещи, как, например, страницу Last.fm в особой вкладке. Успеху *Nightingale* способствуют ровное воспроизведение и хорошее редактирование тэгов. Добавьте сюда превосходный контроль с клавиатуры и его мощные пакеты — конечно, при условии, что ваше оборудование со всем этим справится.



Поддержка и документация

Ибо каждому нужна рука помощи.

Когда *QMPD* сообщил, что «не может соединиться с lastfm», там даже намека не было на необходимость установить *mpdscribble* и отредактировать **/etc/default/mpdscribble**, а затем перезапустить сервис. У *MPD* репутация сложного в настройке плейера, но пусть вас это не пугает. По большей части вы справитесь, следуя инструкциям на сайте *MPD* либо одному из его превосходных онлайн-руководств.

Nightingale, дитя Сети, по щелчку на Help загрузит для вас онлайн-форумы во встроенное окно браузера. Встроенный поиск найдет решение большинства проблем; и когда *Nightingale* отказался запускаться после первой установки, то сообщил, что переключение на US-English часто решает проблемы при запуске... и переключился за нас. Отличная работа.

Справочник по *Banshee* ввели в релиз 1.8 четыре года назад. Это практическое руководство по управлению и воспроизведению музыки, а также по расширению *Banshee* и соединению с переносными плейерами, предлагает полезные советы: скажем, как добыть с Amazon купленные файлы до истечения срока файла **.amz**.

Audacious прост настолько, что почти не нуждается в документации. По большей части будет достаточно списка горячих клавиш на основной странице; а если это не так, то на форумах на сайте вы найдете множество советов и подсказок.

GMusicBrowser предлагает форумы, весьма разумный раздел FAQ и wiki разработчиков, а также удобное руководство по менее очевидным функциям — это вполне адекватный уровень поддержки.

Вердикт

Banshee ★★★★★
GMusicBrowser ★★★★
MPD/QMPD ★★★★
Nightingale ★★★★
Audacious ★★★

» Практичное и емкое руководство Banshee — образцовый справочник такого рода.

Работа с оборудованием

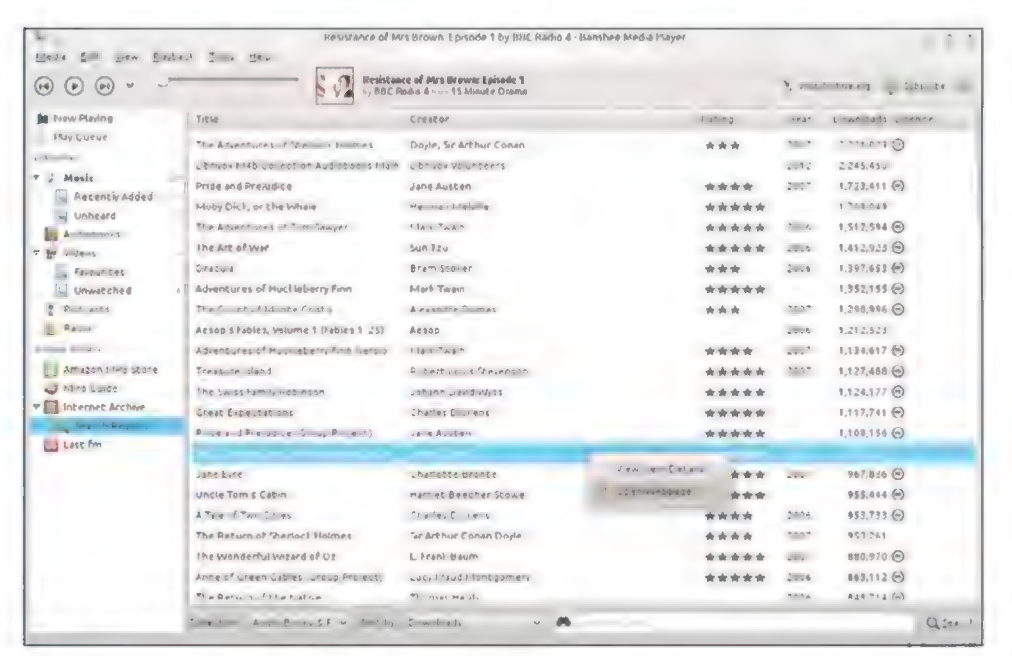
iPod, дистанционное управление и внешние звуковые карты.

Благодаря ALSA, GStreamer и Pulse соединение с USB DAC не должно вызвать проблем, какой бы плейер вы ни взяли, поскольку он распознается как внешняя звуковая карта. И как скептически ни относиться к миру Hi-Fi, где соединительные кабели стоят тысячи фунтов и об оборудовании пишут больше, чем о музыке, все же вложение пары сотен фунтов в USB DAC и достойный усилитель с колонками повлияет на качество музыки гораздо больше, чем выбор лучшего из представленных здесь плейеров (в основном потому, что все они работают весьма достойно).

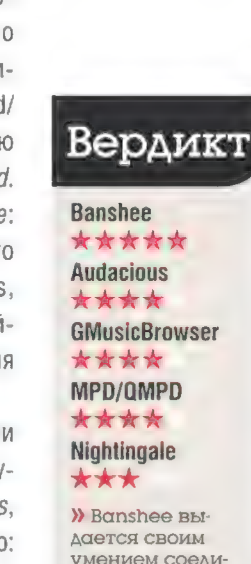
Всеми плейерами можно управлять с телефона через Bluetooth или Wi-Fi, добавив соответствующий пакет Remuco; правда, поддержка *Nightingale* найдется только вне кодовой базы Remuco, в старых пакетах PPA Ubuntu. Но опции дистанционного управления для тех, кому лень вставать с дивана, предлагают все наши подопытные.

Кстати о телефонах: хотя интеграция iPod была и остается главным источником недовольства пользователей Linux, повсеместное распространение телефонов на Android вообще и среди пользователей Linux в частности заметно смягчило эту проблему. Так, *Nightingale* делает импорт и экспорт в iTunes, а управлять iPod/iPhone и т.п. лучше всего через отдельную программу, например, *Podtool* или *Gnupod*. Исключение здесь составляет *Banshee*: у него не только превосходная — и часто обновляемая — поддержка iDevices, но он также контачит со многими устройствами MTP. В *MPD* есть клиент *MPod* для переносных устройств.

Те, кто смотрит на мир сквозь всеми любимый системный монитор *Conky*, будут рады узнать, что он ладит с *Audacious*, *Banshee*, *GMusicBrowser* и *MPD*. Правильно: со всеми, кроме *Nightingale*. Вообще-то большинство плейеров, не представленных здесь, с *Conky* тоже поладят.



› **Даже если вам не пригодится соединение *Banshee* с iPod, вы оцените другие функции.**



Поддержка потокового вещания

Слушаем, что звучит в эфире.

Увы, BBC iPlayer по-прежнему остается недоступным с наших клиентов — если только вы не озаботитесь скриптом регулярного сбора вечно меняющихся адресов m3u для отправки на *MPD* — однако у вас есть опция скачать радиопрограммы BBC скриптом *get_iplayer*, чтобы добавить их в свою библиотеку и прослушать позже.

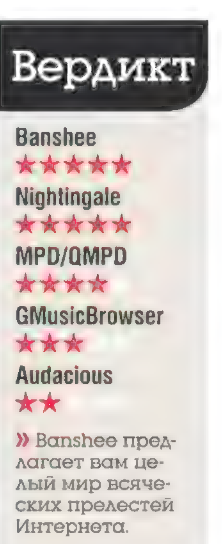
Banshee помещает свои онлайн-ресурсы в левой панели: Amazon, Miro, Internet Archive, Last.fm и т.д. Сторонние плагины добавляют поддержку Jamendo и Magnatune плюс радиостанций с shoutcast.com и xiph.org, а Miro Guide предлагает сотни подкастов.

Nightingale, благодаря своему сетевому наследию, проще всех в плане соединения с Last.fm, и поддерживает Shoutcast и Soundcloud. В *Audacious* способ получения потоков один — опция Add URL в меню, а фишка *Nightingale* — Soundcloud: *Nightingale* — единственный аудиоплейер GNU/Linux, позволяющий с легкостью получать доступ к вашим любимым трекам, и ваш профиль передается в потоке с SoundCloud. Между тем, *QMPDClient* позволит добавить любые интернет-радиостанции, а также станции Shoutcast. Команда **mpc add <URL>** добавляет радиостанции в *MPD*.

Плагин Web Context в *GMusicBrowser* использует MozEmbed или WebKit для отображения страниц Wikipedia и текстов, соответствующих воспроизводимой песне. Помимо этого и поддержки Last.fm, *GMusicBrowser* не использует преимущества Интернета, хотя разработчик проекта Кантен Скюло [Quentin Sculo] говорит, что интернет-радио у него в списке обязательных задач.

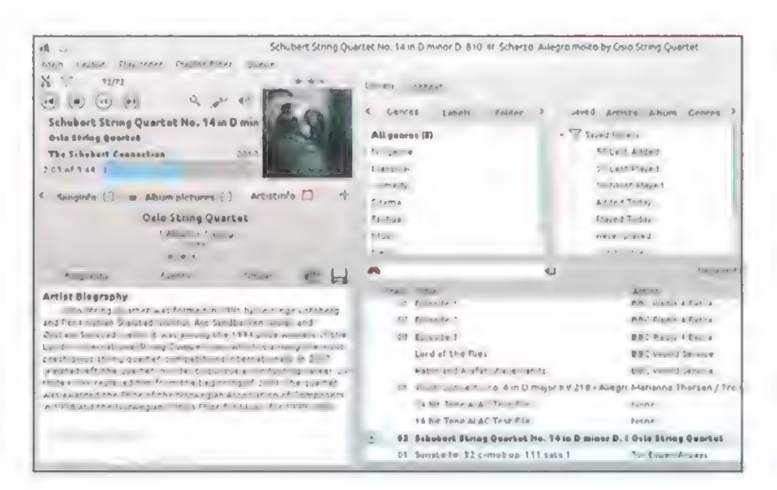


› ***Nightingale*, основанный на Mozilla XULRunner, легко встраивает интернет-сервисы.**



Музыкальные плейеры

Вердикт



› **Причудливый интерфейс *GMusicBrowser* удивительно прост в использовании и сосредоточен на управлении музыкальной коллекцией.**

Определить победителя было нелегко: у всех пятерых есть что предложить. Естественно, все зависит от ваших требований к музыкальному плейеру, способному, в частности, справиться с большой подборкой музыки. Если вас не волнует потоковое вещание, стоит обратить внимание на *Audacious*: им приятно пользоваться, и он хорошо справляется с работой, что редко встречается среди настольного ПО (как это ни грустно).

Nightingale очень мощный. На беду, он ест много больше системных ресурсов, чем другие плейеры нашего теста, а значит, его отнюдь не здорово будет использовать на бедняжке Raspberry Pi. Несмотря на это и на отсутствие интеграции с *Conky* и «родной» с Remuco, *Nightingale* — весьма способный плейер; он справится с большими коллекциями музыки, употребив для управления ими мощь своих расширений.

Любое решение на основе *MPD* потеряет баллы из-за более чем необычной установки и настройки. Тем читателям, которых не огорчает необходимость преодолевать этот барьер, мы рекомендуем *MPD* и один из его множества клиентов, *QMPDClient*, для простой и стабильной работы с большими подборками. Гибкость его сетевой архитектуры клиент–сервер (с клиентами на iPods, в web-браузерах и везде, где только можно себе представить), и разделение сервера (*MPD*), основных функций управления (MPC) и клиентов представляется правильным подходом к выполнению работы — в стиле Unix.

Возвращаясь к более дружелюбным к пользователю приложениям, у нас остается выбор между отлично работающими *Banshee* и *GMusicBrowser*. И вот наш удивительный результат: в *Banshee* куча функций, и он демонстрирует невероятную гибкость, но наименее удобен при работе с большими музыкальными коллекциями, и даже способен при этом спотыкаться и быть нестабильным. Короче говоря, пробуйте сами. *Banshee* может вас и удовлетворить, но он не лишен недостатков. А к более надежному управлению большой подборкой аудиофайлов отлично подогнан *GMusicBrowser*. Он не является лучшим по всем параметрам, но нет ничего, с чем бы он справлялся плохо; и отчасти перегруженный интерфейс для нас был вполне нормальным, даже без перехода на Shimmer или другую тему.

> «К надежному управлению большой подборкой отлично подогнан GMusicBrowser.»



Сайт: gmusicbrowser.org **Лицензия:** GNU GPL **Версия:** 1.1.13

» Предназначен для больших коллекций; странноват, но силен.



Сайт: banshee.fm **Лицензия:** MIT **Версия:** 2.6.2

» Чуть менее мощный, но с массой функций, и очень мил в работе.



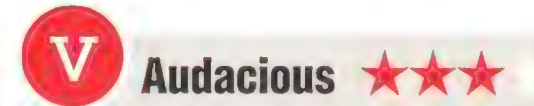
Сайт: musicpd.org **Лицензия:** GPLv2, MPL, BSD **Версия:** 0.18.14/1.2.2

» Подход Unix: необычная настройка и солидные результаты.



Сайт: getnightingale.com **Лицензия:** GPLv2, MPL, BSD **Версия:** 1.12.1

» Изящный и всесторонний, мощный и расширяемый.



Сайт: audacious-media-player.org **Лицензия:** GNU GPL **Версия:** 3.5.1

» Хороший — но в данной компании неконкурентоспособный.

Обратная связь

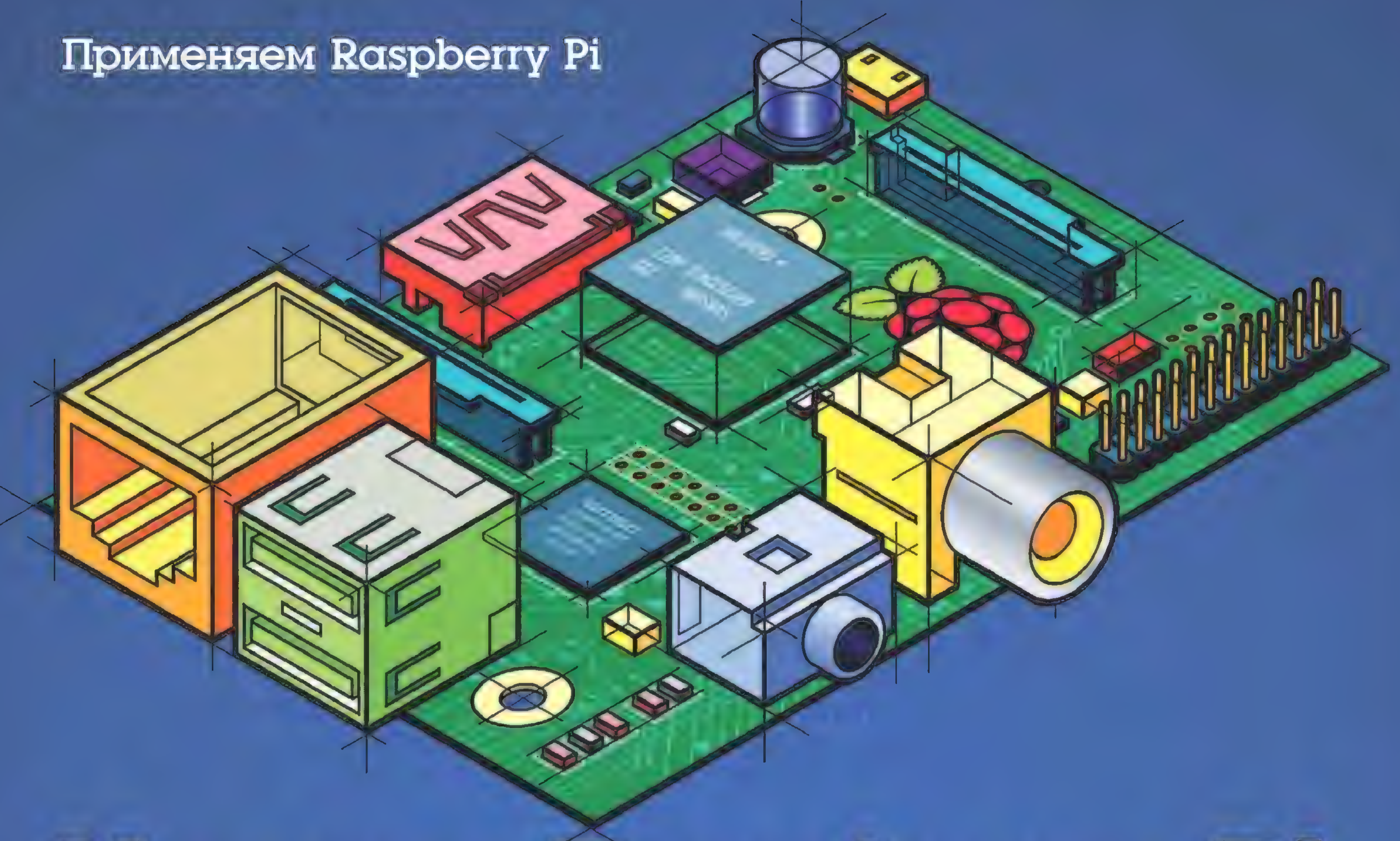
А вы что думаете? Ваш любимый медиа-плейер не попал в наш список? Присылайте нам свои мнения по электронной почте на lxf.letters@futurenet.com.

Рассмотрите также...

Эта область очень перенаселена, и особо стоит упомянуть в ней *Amarok*, который значительно улучшился в прошлых релизах. Также заслуживают упоминания *Quod Libet* с его развитыми тэгами, и более скромный *Qmmp*: ему не хватает функций по сравнению с нашей пятеркой, но он хорошо работает с большими коллекциями. *Qmmp* также спокойно воспроизводит 24-битные треки при частоте сэмплирования 96 кГц или 192 кГц в среде стерео или 5.1 — но не треки DSD. Мы также большие поклонники *Clementine* с его превосходным управлением плейлистами во вкладках и потоковыми сервисами.

В наши дни имеют место пересечения между аудио- и медиа-плейерами, и хотя последние не обязательно предназначены для работы с большими подборками аудиофайлов в легкой и изящной манере, *UMPlayer* — хороший пример, и он показал хорошие результаты в Сравнении медиа-плейеров (см. **LXF187**, стр. 28), как и *SMPlayer*.

Если вам нужна гибкость схемы клиент–сервер и выбор интерфейсов, но без сложности *MPD*, то *XMMS2* вам это предложит, а заодно — отличное ровное воспроизведение.



Хакинг Raspberry Pi

Следуйте за мастерами свободного ПО, и Маянк Шарма покажет вам, как овладеть премудростями Raspberry Pi, научившись его программировать.

Raspberry Pi был разработан как устройство для обучения детей. В Raspberry Pi Foundation создали полнофункциональный компьютер без всяких излишеств, чтобы компьютерные технологии стали доступны тем детям, которые хотят научиться программировать, но не имеют средств на приобретение готового оборудования. Однако это устройство завоевало популярность у программистов, которые применяли его весьма творчески, выведя его далеко за рамки изначальной аудитории, планировавшейся Эбеном Аптоном [Eben Upton], Робом Маллинзом [Rob Mullins], Джеком Лэнгом [Jack Lang] и Аланом Майкрофтом [Alan Mycroft] при обсуждении этой идеи в Компьютерной лаборатории Кембриджского университета.

Но из-за этого возникло недопонимание. Многие считают, что Pi работает либо в начальной школе, либо в руках профессионалов, способных подключить его к радиоуправляемому автомобилю или к плюшевым мишкам-космонавтам. Это тоже правда, но Pi является превосходным инструментом также и для использования в областях, расположенных между этими двумя крайностями.

А теперь не удивляйтесь, если мы вам скажем, что эксперименты с Pi — это отличный способ изучить Linux и программирование. И мы не единственные, кто это утверждает. Мы уже подробно рассказывали о новой программе школьного образования в Великобритании и о том, как она поощряет использование Raspberry Pi на каждом из основных своих этапов (см. **LXF189**, стр. 60).

> «Raspberry Pi был разработан как устройство для обучения детей программированию...»

На Этапе 1 дети в возрасте 5–7 лет пишут и тестируют простые программы на разных устройствах, например, на планшетах или Raspberry Pi. Затем идет Этап 2, когда дети 7–11 лет знакомятся со Scratch, установленным по умолчанию в официальном дистрибутиве Pi, Raspbian. На Этапе 3 дети 11–14 лет изучают текстовые языки типа Python для управления электронными устройствами. Pi — чудесная платформа для такого применения благодаря его портам GPIO, к которым легко подключить устройства вроде Pibrella. Наконец, на Этапе 4 ученики 14–16 лет могут заниматься с отличными дополнительными комплектами для Pi, развивая свои компьютерные навыки и знания.

Данной статьей мы поможем вам освоить новые навыки программирования с Raspberry Pi. Все, что вам нужно — наличие Pi под рукой. Употребите *NOOBS* для подготовки SD-карты для Pi. Скачайте его или возьмите с диска, распакуйте и скопируйте полученное содержимое на отформатированную SD-карту, и вот у вас уже все готово для работы. В этой статье рассматриваются очень практичные повседневные проекты, которые может реализовать каждый, независимо от уровня знаний. По мере выполнения этих 10 заданий вы познакомитесь с рядом широко используемых хитростей ремесла...

Навыки: Безопасность, API Twitter

Задача №1: Настроим камеру слежения

Запуск официального модуля камеры Raspberry Pi открыл целый мир возможностей. Программисты, уже применявшие USB-камеры, получили крошечные камеры Full HD, идеально подходящие для слежения. В сочетании с вычислительными способностями Pi это позволяет отправлять сделанные снимки в свой Twitter.

Прежде чем присоединять камеру, найдите ее интерфейс Camera Serial Interface (рядом с портом Ethernet) и осторожно потяните вывод вверх. Теперь протолкните кабельный шлейф модуля камеры в слот — так, чтобы серебряные контакты на кабеле смотрели в противоположную от порта Ethernet сторону. Помните, что нельзя проталкивать кабель слишком глубоко. Теперь удерживайте его на месте одной рукой и проталкивайте вывод CSI вниз другой рукой, чтобы шлейф зафиксировался.

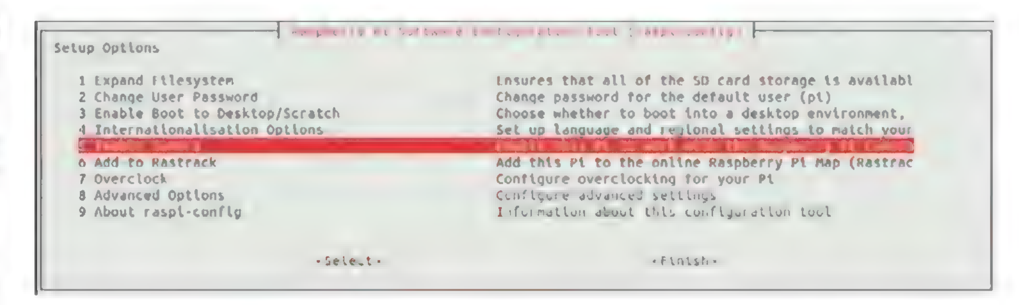
Итак, оборудование установлено; пора настроить программы. Загрузитесь в Raspbian и вызовите инструмент настройки вашего Pi с помощью **sudo raspi-config**. Прокрутите список до пункта Enable Camera. Вам нужно подтвердить свой выбор и перезапустить Pi. После перезапуска вы сможете использовать хорошо документированные команды *raspistill* и *raspivid*, чтобы делать фотографии и видеосъемку.

Для захвата движения мы задействуем легковесный скрипт на Python для обнаружения движения, написанный членами сообщества Raspberry Pi. Скрипт использует Python Imaging Library, это библиотека для анализа и обработки изображений. Вы можете установить ее командой **sudo apt-get install python-imaging-tk**. Также создайте директорию с именем **picam** в своей домашней директории, чтобы скрипт хранил там изображения: **mkdir ~/picam**. Теперь возьмите скрипт и сделайте его исполняемым с помощью **wget -c http://pastebin.com/raw.php?i=yH7JHz9w -O picam.py** и **chmod +x picam.py**.

Когда вы запустите скрипт с помощью **./picam.py**, он включит красный светодиод на камере Pi и примется делать снимки с низким разрешением. Затем он начнет определять движение, сравнивая пиксели в двух последовательных снимках. Определив появление различий, скрипт запустит снятие изображений с высоким разрешением. Скрипт очень эффективен, и он автоматически удаляет сделанные им для сравнения снимки с низким разрешением и сохраняет только снимки с высоким разрешением, где зафиксировано движение.



› Дистрибутив Kali Linux разработан специально для тестирования на проникновение, и есть версия для Raspberry Pi.



› Модуль камеры Raspberry Pi весит всего 9 граммов, но умеет делать снимки с разрешением 2,592×1,944 и Full HD 1080p видео 30FPS и 720p видео 60 FPS.

Чтобы запустить скрипт при загрузке, вам нужен init-скрипт, запускающий скрипт *picam.py* и останавливающий его перед выключением Raspberry Pi. И опять-таки сообщество сделало за вас всю подготовительную работу. Просто скачайте их скрипт — **wget http://pastebin.com/raw.php?i=AfqbjQrb -O picam_init** — и переместите его куда требуется: **sudo mv ~/picam_init /etc/init.d/picam**, а затем сделайте его исполняемым: **sudo chmod +x /etc/init.d/picam**. И, наконец, покажите этот скрипт системе загрузки с помощью **sudo update-rc.d picam defaults**...

Теперь скрипт будет запускаться и выключаться вместе с Raspberry Pi. Вы также можете контролировать его вручную, как любой другой демон. Например, **/etc/init.d/picam stop** остановит скрипт, а **/etc/init.d/picam start** запустит его.

Отошлите в Twitter

Теперь настроим новую учетную запись в Twitter и дадим Pi команду размещать там изображения. Проверьте, чтобы учетная запись была частной. Начните с установки pip, менеджера пакетов Python, для установки библиотек Python, посредством **sudo apt-get install python-pip**. Затем установите Twython, надстройку Python для API Twitter, с помощью **sudo pip install twython**. Примечание: чтобы использовать Twython, вам нужна учетная запись разработчика Twitter. Перейдите на https://apps.twitter.com и введите всю необходимую информацию по новой учетной записи. На открывшейся странице щелкните по Create New App и в предоставленном поле введите его название и описание. Пропустите поле Callback URL, прокрутите страницу вниз и создайте приложение.

Изначально приложение создается с разрешениями только на чтение. Щелкните по приложению, перейдите во вкладку Permissions и переключите радиокнопку Read and Write. Затем перейдите во вкладку API Keys и нажмите на кнопку Create My Access Token. Отметьте переменные API Key, API Secret, Access Token и Access Token Secret, перечисленные на этой странице.

Затем скачайте модифицированный скрипт *picam*, изменения в котором касаются размещения изображений в Twitter, через https://raw.githubusercontent.com/ghalfacree — зайдите в bash-scripts и поищите *picam.py*.

Откройте этот новый файл **picam.py** в текстовом редакторе и введите четыре бита информации, взятые из Twitter в поле в верху скрипта.

Закончив, сделайте его исполняемым с помощью **chmod +x picam.py**. Теперь, когда вы запустите скрипт и он определит движение, помимо съемки и сохранения изображения на SD-карте, Pi также отправит его на вашу частную учетную запись Twitter, и оно появится в вашей ленте Twitter.

Навыки: Работа в сети, взаимодействие, управление данными

Задача № 2: Сервер доступа к файлам

Возможность делиться данными на вашем Raspberry Pi и получать доступ к ним с других компьютеров очень удобна. Например, если Pi работает постоянно действующим устройством для скачивания, нужно же будет перемещать скачанные данные с Pi. Эти данные могут быть или на SD-карте, или на присоединенном USB-диске. С помощью ПО *Samba* — это реализация сетевого протокола SMB/CIFS — вы можете использовать свой Raspberry Pi как сетевое хранилище (Network Attached Storage, NAS) и легко получать с компьютеров в вашей сети доступ к диску USB, присоединенному к Pi.

Команда **sudo apt-get install samba samba-common-bin** найдет нужное ПО. Теперь подключите USB-диск к Pi, и он автоматически смонтируется в папку **/media**. Предположим, USB-диск смонтирован в **/media/usb**. Теперь надо настроить *Samba* так, чтобы диск выдавал сети доступ. Для этого сначала надо добавить пользователя *Samba* с именем pi. Введите команду **sudo smbpasswd -a pi** и затем введите пароль.

› *Remmina* предлагает очень удобный интерфейс и автоматически прокручивается, когда мышь упирается в край экрана.

Далее откройте файл настройки Samba (**/etc/samba/smb.conf**) в текстовом редакторе. Если вы хотите получить доступ к Pi с компьютера с Windows, разместите строку **workgroup = WORKGROUP** где-то в начале файла **smb.conf**, проставив название вашей рабочей группы Windows. Далее внизу файла найдите строку **# security = user** и удалите #, чтобы раскомментировать строку и включить безопасность. И, наконец, прокрутите до конца файла и добавьте следующие строки:

```
[USB]
path = /media/usb
comment = USB NAS Drive
valid users = pi
writeable = yes
browseable = yes
create mask = 0777
public = yes
```

Сохраните файл и затем перезапустите *Samba* командой **sudo /etc/init.d/samba restart**.

Вот и все. Теперь вы должны быть в состоянии получить доступ к диску USB, присоединенному к Pi с любого другого компьютера в сети.

Задача № 3: Pi как тонкий клиент

Тонкий клиент — это компьютер, который при решении задач полагается на другие, более мощные компьютеры, передавая им вычислительную нагрузку, а сам только представляет результаты. Благодаря энергоэффективной и бесшумной работе, Raspberry Pi является естественным тонким клиентом. Тонкий клиент использует удаленные протоколы рабочего стола для общения с мощным удаленным рабочим столом.

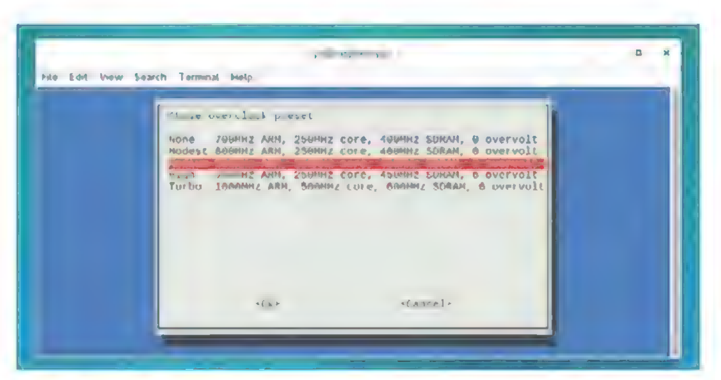
Для данной задачи понадобится мощный компьютер, который будет служить удаленным сервером, и Raspberry Pi, который будет тонким клиентом. Наш сервер рабочего стола работает на Ubuntu

Разгоните Pi

Хотя процессор Pi на 700 МГц вполне годится для выполнения наших задач, со временем вы решите, что неплохо бы выжать из него побольше. И вот хорошая новость: такое возможно! Процессор Pi BCM2835 способен превзойти свою скорость по умолчанию...

Однако помните, что за этот подвиг надо платить — процессор будет потреблять больше энергии, нагреваться сильнее, и может не протянуть так же долго, как обычный Pi, работающий на скорости по умолчанию. Производительность Pi можно изменить вручную, но самый безопасный способ сделать это — выбрать предварительно заданные настройки разгона в инструменте *raspi-config*, что к тому же не отменяет гарантии.

Настройки в *raspi-config* считаются безопасными для большинства устройств Pi. Чтобы разогнать свое устройство, запустите *raspi-config*, прокрутите вниз опцию Overclock и подтвердите, что хотите продолжить. Вам будет представлен список преднастроенных скоростей разгона — выбирайте любую из них. Когда вы сделаете выбор, меню перезагрузит Raspberry Pi с новой скоростью.



› Если Pi не загружается с настройками разгона, при загрузке придержите Shift, это установит штатную скорость.

с предустановленным сервером удаленного рабочего стола *Vino*, доступ к которому можно получить через приложение *Desktop Sharing*. Функциональность удаленного рабочего стола по умолчанию отключена. Чтобы ее включить, запустите приложение *Desktop Sharing* и отметьте окно Allow other user to view your desktop. Это также включает вторую кнопку-флажок, позволяющую подсоединенным пользователям управлять сервером Ubuntu.

Включена также первая опция в разделе Security, которая вынуждает вас утверждать каждый запрос на соединение. Однако для более ровной работы вам захочется отключить эту опцию, убрав галочку. Вместо этого включите следующую кнопку-флажок, по которой пользователю будет предлагаться ввести пароль, и введите сильный пароль в показанном поле. Закончив, нажмите на кнопку Close, чтобы сохранить все изменения. Если вы используете другой дистрибутив, поищите в Сети и установите поверх этого дистрибутива сервер VNC, такой, как *Vino* или *Krfb*.

Если вы выберете *Vino* от Ubuntu, придется внести еще одно дополнительное изменение, потому что *Vino* был изменен так, чтобы по умолчанию требовать шифрования, но поддерживает только старые методы шифрования, которые уже не столь распространены. Запустите терминал на сервере Ubuntu и измените настройки безопасности *Vino* с помощью **gsettings set org.gnome.Vino require-encryption false**. Следуйте решению задачи, и, если вы сможете соединиться с Pi, вернитесь на Ubuntu Server и сделайте настройки постоянными: установите *dconf-editor* по **sudo apt-get install dconf-editor** и, перейдя в org > gnome > desktop > remote-access, уберите галочку с настройки require-encryption.

Теперь подготовим тонкий клиент Pi. Для этой задачи мы будем использовать легковесный клиент *Remmina*, устанавливаемый командой **sudo apt-get install remmina**. После установки он попадет в меню Internet. Запустите клиент и щелкните по новому значку, чтобы настроить соединение с сервером.

В открывшемся окне дайте соединению имя и выберите VNC из выпадающего меню Protocol, а в поле Server укажите IP-адрес сервера. Качество изображения можно улучшить, выбрав бо́льшую глубину цвета и лучшее качество из соответствующих выпадающих списков. По завершении щелкните на Connect. *Remmina* установит соединение с Ubuntu Server и предложит вам ввести заданный вами пароль, после чего разрешит вам принять на себя управление удаленным рабочим столом из Pi.

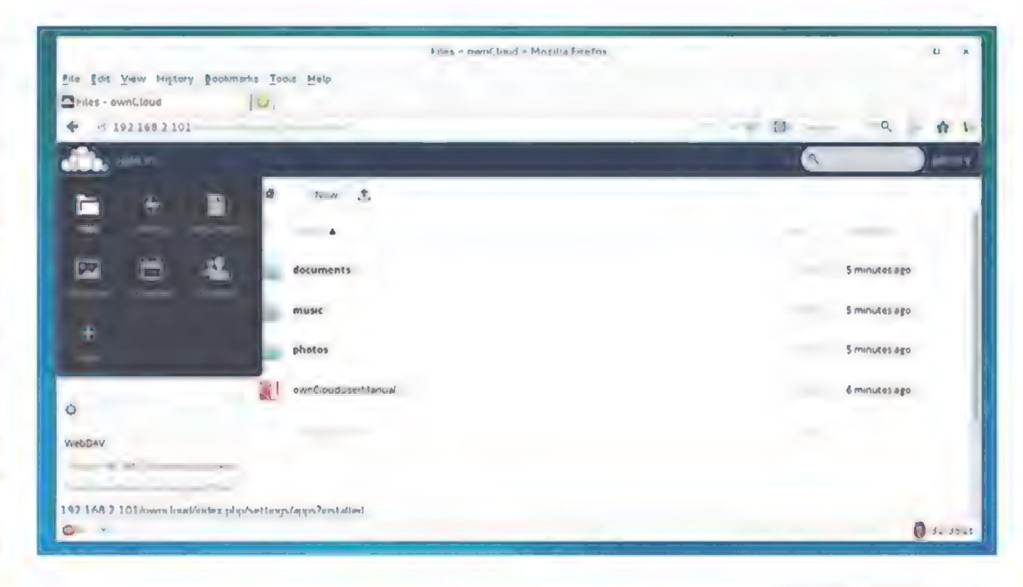
Задача № 4: Разместите свой ownCloud

Если вам нужен полный контроль над своими данными, не стоит выбрасывать деньги на сервис, который иной раз даже хранит данные на серверах за пределами вашей юрисдикции. Вместо этого потратьтесь на Pi и емкий USB-диск и разместите там личный и защищенный облачный сервис с помощью *ownCloud*.

OwnCloud позволяет синхронизировать и делиться данными и получать доступ к ним с любого устройства, соединенного с Интернетом. Для большей безопасности *ownCloud* может также зашифровать ваши файлы. Программа умеет работать с файлами в разных форматах, и в нее встроены просмотрщик изображений галереи и музыкальный плейер. Интересная функция *ownCloud* — присвоение версий файлам; благодаря ей сервер отслеживает все изменения в каждом файле, и вы можете вернуться к более старой версии за один щелчок.

Подобно другим онлайн-сервисам облачного хранения, вы можете синхронизировать файлы в *ownCloud* или с помощью браузера, или с помощью настольного клиента в Windows, Mac и Linux, а также мобильных клиентов для устройств Android и iOS.

OwnCloud работает на web-сервере *Apache*, и ему также нужен сервер базы данных. Хотя он может работать на *MySQL*, для данной задачи мы взяли легковесный сервер *SQLite*. Все нужные компоненты установит **sudo apt-get install apache2 php5 php5-gd php5-sqlite curl libcurl3 php5-curl**.



› *ownCloud 7* — **важный шаг от более ранних версий, с лучше организованным рабочим процессом.**

Теперь перейдите на https://owncloud.org и скачайте tar-архив самой свежей версии. Распакуйте его командой **tar xjvf owncloud-7.0.2.tar.bz2** и перейдите в полученной папке в корень сервера *Apache* по **sudo mv owncloud /var/www**. Затем убедитесь, что новые файлы имеют правильные разрешения для их нового местоположения, с помощью **sudo chown -R www-data:wwwdata/var/www/owncloud**.

Для корректной работы *ownCloud* потребуется включить кое-какие модули *Apache*. В терминале введите **sudo a2enmod headers rewrite env** и перезапустите *Apache* через **sudo service apache2 restart**. Чтобы настроить *ownCloud*, запустите web-браузер и перейдите в экземпляр установки *ownCloud* на localhost/owncloud. Поскольку это новая инсталляция, вам предложат создать учетную запись администратора *ownCloud*.

> «Если нужен полный контроль над данными, берите Pi и USB-диск и разместите облако!»

Отныне смело заходите на сервер *ownCloud* как администратор и начинайте выгрузку файлов, которыми хотите поделиться. Но перед этим следует подправить файл конфигурации PHP, если вы собираетесь выгружать файлы размером более 2 МБ. Для этого откройте в текстовом редакторе файл конфигурации PHP, **php.ini**, размещенный в **/etc/php5/apache2**. Найдите переменные upload_max_filesize и post_max_size и измените их значение с 2M на что-то вроде 500M или 1G.

Теперь у вас все готово, чтобы загружать данные на свой сервер *ownCloud* через web-интерфейс. Вы также можете взаимодействовать с сервером *ownCloud* через протокол WebDAV. В области размещения в менеджере файлов Files нажмите на Ctrl+L, чтобы включить область размещения. Здесь вы можете указать на свой сервер *ownCloud*, например: dav://localhost/owncloud/remote.php/webdav.

После аутентификации все, что хранится на *ownCloud*, смонтируется, и вы сможете взаимодействовать с ним, как с обычной папкой.

Чтобы поделиться загруженными файлами, перейдите в раздел Files web-интерфейса и затем — в файл, которым хотите поделиться. У вас отобразится несколько опций, в том числе Share — она позволяет выбрать, с кем вы хотите делиться и даете ли вы им разрешение редактировать и удалять файлы.

Можно делиться и с теми, кого нет на вашем сервере *ownCloud*. По щелчку на Share с флажком ссылки *ownCloud* отобразит ссылку на объект, который вы хотите предоставить кому-то в Интернете. Есть опции защиты ссылки паролем и установки даты истечения срока действия.

Навыки: вещание, эмуляция и GPIO

Задача № 5: Вещаем потоком из Сети

Нас окружает музыка. Помимо свободных от DRM треков на жестком диске, у вас, вероятно, есть сотни треков на Spotify или Google Play. Вы можете собрать их все вместе с помощью дистрибутива PiMusicBox, который превратит Raspberry Pi в отличный музыкальный плейер.

PiMusicBox умеет транслировать музыку через колонки, присоединенные к разъему для наушников Pi, а также через порты HDMI и USB. Вам остается только установить дистрибутив, подключить к Pi какие-нибудь колонки, ввести имя пользователя и пароль вашей учетной записи, и пускай воспроизводит. Потом вы сможете контролировать свой накачанный Pi с любого компьютера в сети, и даже с устройства Android.

Начните со скачивания сжатого образа для дистрибутива PiMusicBox с www.pimusicbox.com. Распакуйте скачанный ZIP-файл и затем поместите файл образа **.img** на SD-карту командой **dd**, например: **sudo dd if=musicbox0.5.img of=/dev/sdd**. Не забудьте заменить **/d/ev/sdd** на местоположение вашей SD-карты.

Если вы используете для соединения Pi с Интернетом порт Ethernet, можете загрузить Pi со вновь созданной SD-карты. Но если вы используете беспроводную карту, нужно будет отредактировать файл настройки своего дистрибутива и вручную указать ему путь к вашему беспроводному роутеру. Зайдите на вновь созданную SD-карту из обычного дистрибутива, перейдите в папку **config** и откройте файл **settings.ini** в текстовом редакторе. Около начала файла вы увидите две переменные, WIFI_NETWORK и WIFI_PASSWORD. Припишите значения этих двух переменных, соответствующие вашей сети, и сохраните файл. Отметим, что PiMusicBox будет работать только с WPA2-защищенными беспроводными сетями.

Проделав это, загрузите Pi с настроенной SD-карты. При первой загрузке дистрибутив изменит размер файловой системы, чтобы занять всю карту целиком, и перезапустится автоматически. Если у вас к Pi подсоединен монитор, можете следовать процессу загрузки; в ином случае подождите одну-две минуты, запустите браузер на любом компьютере своей сети и перейдите в http://musicbox.local. Если вы никуда не перешли, то укажите в своем браузере IP адрес Pi.

Интерфейс по умолчанию PiMusicBox довольно пуст, поскольку вы пока что не настроили ни одного музыкального источника.

› PiMusicBox основан на музыкальном сервере *Mopidy*, написанном на Python.

Чтобы это исправить, нажмите на ссылку Setting на панели навигации слева. Вы окажетесь на странице, где можно индивидуально включать и настраивать все сервисы поддержки потоков, от премиальных, типа Spotify и Google Music, до бесплатных, таких как The Internet Archive, Soma FM и прочих. Вы также можете управлять другими настройками с той же страницы. Например, настройка Audio позволяет переключать устройства воспроизведения аудио.

Дистрибутив PiMusicBox имеет рабочую настройку *Samba* и должен появиться в разделе Network в файловом менеджере всех ОС. Каталог *Samba* в дистрибутиве содержит всего одну папку — **Music**. Вы можете поместить в эту папку любой аудиофайл, и он будет перемещен на SD-карту. Когда бы вы ни перезапустили Pi, дистрибутив просканирует любую новую музыку при загрузке. Затем можно будет просмотреть ее и воспроизвести эти файлы из web-интерфейса дистрибутива.

Вы также можете воспроизводить музыку через любые программы, которые поддерживают *Music Player Daemon* (*MPD*), например, приложение *MPDroid* для Android. Для соединения запустите приложение, и при первом запуске введите IP-адрес Pi в поле имени хоста в появившемся мастере.

Задача № 6: Трансляция аудио

Потоковое воспроизведение музыки в Pi — это одно. А если вы хотите передать музыку на другие устройства? И раз уж вы за это взялись, почему бы не запустить собственную радиостанцию? Как выясняется, это можно сделать без особых хлопот.

Помимо знакомых портов Audio, Ethernet, HDMI и USB, Raspberry Pi также снабжен интерфейсами, предназначенными для более непосредственного соединения с другими чипами и модулями. «Порты» General Purpose Input/Output (GPIO) — это игольчатые контакты, расположенные двумя параллельными рядами (26 на плате Model B и 40 на B+).

Эти интерфейсы не являются plug-and-play, но их можно контролировать через программы. Программисты из Code Club написали программу, чтобы употребить контакты, предназначенные для генерирования синхронизирующих сигналов широкого спектра, на то, чтобы вместо этого выводить сигналы FM Radio. Чтобы передать на удивление сильный сигнал FM, все, что вам нужно — присоединить проволоку к контакту GPIO 4. Даже без проволоки сигнал FM, переданный Pi, будет принят расположенными поблизости приемниками FM.

Включите Pi и выведите терминал. Теперь возьмите код, написанный в Code Club, и распакуйте его с помощью Wget..:

```
http://omattos.com/pifm.tar.gz
mkdir ~/pifm
tar zxvf pifm.tar.gz -C ~/pifm
```

Tar-архив распакует шесть файлов. Как ни странно, этого и достаточно. Теперь вы можете транслировать включенный файл **sound.wav** с помощью **sudo ./pifm sound.wav 101.2**. Теперь найдите приемник FM, настройте его на FM 101.2, и вы услышите тему из «Звездных войн». На самом деле, частоту трансляции можно варьировать между 88 МГц и 108 МГц, просто приписав частоту канала в конце команды.

Можно воспроизводить и другие аудиофайлы, но только 16-битные 22,050 кГц моно и только в формате WAV — что

выглядит ограничением, но на самом деле это не так, благодаря блестящему аудиоредактору *SoX sound exchange*. Мы будем использовать этот замечательный инструментик для обработки любого MP3-файла независимо от его кодировки, на лету конвертируя его в правильный WAV-файл.

Начните с установки аудиоредактора и его зависимостей через **sudo apt-get install sox libsox-fmt-all**. Закончив, введите следующую команду, заменив **SomeSong.mp3** на имя MP3-файла, который вы хотите воспроизвести:

```
sox -t mp3 SomeSong.mp3 -t wav -r 22050 -c 1 - | sudo ./pifm
 - 101.2
```

Первая часть команды конвертирует MP3-файл в WAV-файл, изменяет его частоту аудиосэмплирования на 22050 Гц и микширует трек в моно. Затем конвертированный трек отправляется в стандартный вывод (обозначенный дефисом), который затем перенаправляется вертикальной чертой (|) на стандартный ввод команды *pifm*.

Единственное отличие команды *pifm* в вышеприведенном примере то, что вместо указания имени файла для трансляции мы просим скрипт транслировать информацию во стандартного ввода. Если ваш FM-приемник все еще настроен на частоту 101.2, вы теперь должны услышать свой MP3.

Со своим *SoX* можно буквально творить чудеса: например, использовать его для трансляции своих любимых потоков в режиме live из Интернета. Команда **sox -t mp3 http://www.tuxradar.com/files/podcast/tuxradar_s06e02.mp3 -t wav -r 22050 -c 1 — | sudo ./pifm — 101.2** будет транслировать подкаст TuxRadar. Единственное различие между этой командой и предыдущим примером в том, что вместо указания на локальный MP3 вы указываете на MP3, размещенный онлайн.

Задача №7: Эмуляция винтажных игр

Игры не всегда были графическими шедеврами. Раньше у разработчиков было куда меньше ресурсов, и вместо графического совершенства они в большей степени полагались на сам игровой процесс. Именно поэтому винтажные игры при всей рудиментарности их графики до сих пор популярны у геймеров всех возрастов — даже в наши дни ничто не сравнится с классической игрой, даже на Steam или Live.

Простейший способ начать играть в винтажные игры на Raspberry Pi — установить дистибутив RetroPie: в нем есть эмуляторы. Вы можете установить его на существующую инсталляцию Raspbian или запустить его как отдельный дистрибутив. Прежде чем скачивать скрипт установки, найдите его зависимости командой **sudo apt-get install git dialog**. Затем скачайте самый свежий скрипт установки:

```
git clone git://github.com/petrockblog/RetroPie-Setup.git
```

Перейдите во вновь клонированную директорию и запустите скрипт с помощью **cd RetroPie-Setup && sudo ./retropie_setup.sh**. Скрипт скачает недостающие зависимости и представит вам меню с несколькими опциями.

Первая опция скачивает и устанавливает прекомпилированные бинарники всех популярных поддерживаемых платформ. Вторая скачивает самый новый код для каждого приложения и компилирует их на Pi. При выборе этой опции запаситесь терпением, поскольку это может занять чуть ли не целый день. Если терпения у вас не настолько много, можно с тем же успехом скачать файл образа и переместить его на пустую SD-карту по **dd**.

RetroPie использует графический интерфейс под названием *EmulationStation*, котоый позволяет управлять установленными

> «Чтобы играть в игры, вы, естественно, должны их иметь...»

эмуляторами. Образ RetroPie автоматически запускает этот интерфейс, но если вы установите RetroPie поверх существующей инсталляции Raspbian, его придется запускать вручную через команду *emulationstation*.

Когда все будет готово, вы увидите экран настройки контроллера, где можно настроить ваш USB-контроллер игры. Чтобы играть в игры, вы, естественно, должны их иметь. ROM можно сделать из ваших копий этих старых игр, или поискать позабытые прошивки онлайн. Некоторые компании, такие, как iD software, сделали *Doom* с открытым кодом, выведя его в общественный домен. Найдя игры, скопируйте их в соответствующую им подпапку эмулятора внутри папки **roms/** на SD-карте.



› *EmulationStation* **будет показывать только эмуляторы, в которые добавлены ROM.**

Пожертвуйте излишки ресурсов

Если ваш Pi занят не постоянно, вы можете пожертвовать простаивающую мощь процессора на благое дело. *BOINC* — добровольный компьютерный сервис, и он использует пожертвованные ресурсы в разных проектах, от сворачивания белка для медицинских наук до поиска внеземного разума.

Перед установкой *BOINC* настройте Raspberry Pi так, чтобы урезать собственное потребление ресурсов, вызвав **sudo raspi-config**. Перейдите в Advanced Options > Memory Split и выделите минимально возможный объем памяти для GPU, который, как вы увидите, будет равен 16 МБ. Затем установите *BOINC* через **sudo apt-get install boincmanager boinc-client**.

Закончив, запустите *BOINC Manager* из меню Others. Вам предложат выбор из более чем 30 проектов. Некоторые проекты, возможно, предупредят вас, что для Pi у них подходящей работы нет. Вы все равно можете их добавить, но все же лучше выбрать проекты, с которыми Pi справляется удачнее, например, проект Collatz, который пытается опровергнуть гипотезу Коллатца [Collatz Conjecture].



› **Чтобы настроить для *BOINC* ресурсы Pi, сначала включите** Advance View, **а затем перейдите в** Tools > Computing preferences.

Навыки: Python, eSpeak

Задача № 8: Пусть ваш Pi заговорит

Программные синтезаторы речи всегда были популярны для, так сказать, очеловечивания компьютерных устройств. И, конечно, они играют крайне важную роль в обеспечении доступа к устройству слабовидящим пользователям.

Вы можете использовать Raspberry Pi для преобразования команд из текста в голос, благодаря мощной библиотеке *eSpeak* в репозитории Raspbian. Имеется также модуль, позволяющий использовать *eSpeak* в Python и выполнять задачи по автоматизации. Запустите терминал и используйте команду **sudo aptget install espeak python-espeak**, чтобы найти библиотеку и нужные модули Python.

Использовать библиотеку *eSpeak* очень просто. Введите в терминале **espeak "Hello! How are you doing, today?"** [Привет! Как дела?], и библиотека, используя свои настройки по умолчанию, озвучит текст в кавычках. Затем вы сможете настроить речь библиотеки *eSpeak* с помощью множества переключателей командной строки. Например, команда **espeak -ven+f2 -s140 "Aren't you a little short for a storm-trooper"** [А не малорослый ли вы для штурмовика?] озвучит сообщение в более замедленном темпе и женским голосом. Помимо установленного по умолчанию английского, *eSpeak* умеет говорить с американским и шотландским акцентом, а также и на других языках. Команда **espeak -voices** перечислит все имеющиеся голосовые файлы.

Конвертировать текст в речь с помощью модуля Python *eSpeak* весьма несложно. Вызовите интерактивную оболочку Python командой **python** и введите следующие строки...

```
from espeak import espeak
espeak.synth("How ya doin'?")
```

Теперь давайте используем библиотеку *eSpeak* внутри скрипта Python, чтобы распечатать и произнести вслух имена всех ваших читателей в Twitter. Следуйте инструкциям в первой задаче, чтобы перейти в API Twitter и получить маркеры доступа и их секреты и поместить их между одиночных кавычек в коде ниже:

```
import time
from twython import Twython
from espeak import espeak
api_token = ' '
api_secret = ' '
access_token = ' '
access_token_secret = ' '
twitter = Twython (api_token, api_secret, access_token,
access_token_secret)
next_cursor=-1
while (next_cursor):
  search = twitter.get_followers_list(screen_
name='geekybodhi',cursor=next_cursor)
  for result in search['users']:
    print result["name"]
    their_name= result["name"]
  espeak.synth(their_name)
  time.sleep(2)
  next_cursor = search["next_cursor"]
```

В вышеприведенном коде мы соединились с нашей учетной записью в Twitter и получили список наших читателей. Мы использовали технологию, известную как курсирование [cursoring], для разделения объемных результатов (перечня подписчиков) на страницы и затем перемещения по ним (с помощью **next_cursor**). Цикл For будет повторяться, пока не исчерпает список. Для каждого подписчика цикл будет печатать имя, сохранять его в переменной (**their_name**) и передавать библиотеке *eSpeak*, которая озвучит его. Затем она несколько секунд подождет и перейдет к следующему подписчику.

```
pi@raspberrypi: ~/jasper/client
pi@raspberrypi ~/jasper/client $ python populate.py
Welcome to the profile populator. If, at any step, you'd prefer not to enter the
 requested information, just hit 'Enter' with a blank field to continue.
First name: Mayank
Last name: Sharma

Jasper uses your Gmail to send notifications. Alternatively, you can skip this s
tep (or just fill in the email address if you want to receive email notification
s) and setup a Mailgun account, as at http://jasperproject.github.io/documentati
on/software/#mailgun.

Gmail address: geekybodhi@gmail.com
```

› **Вы можете настроить и расширить *Jasper*, добавив собственные команды.**

Задача № 9: Управление Pi голосом

У Apple есть *Siri*, у Google — *Google Now*, а у Microsoft — *Cortana*. Если вы полагаете, что у Raspberry Pi нет собственного цифрового помощника, значит, вы не слыхали про *Jasper*. В отличие от вышеупомянутых помощников, у *Jasper* открытый код.

Чтобы услышать результаты работы *Jasper*, к аудиопорту придется присоединить колонки. Однако, поскольку у Pi нет микрофона, вам придется найти USB-микрофон, который ладит с Pi. Если в вас есть авантюрная жилка, можете даже попробовать подключить web-камеру USB — в некоторых из них предусмотрены микрофоны. Присоединив микрофон, убедитесь, что Pi распознает его как устройство для записи, с помощью команды **arecord -l command**.

Хотя вы можете установить *Jasper* поверх существующей установки Raspbian, мы рекомендуем использовать образ SD-карты. Скачайте tar-архив с сайта (http://jasperproject.github.io) и распакуйте образ из него командой **tar zxvf jasper-disk-image.tar.gz**. Затем, предположив, что **/dev/sdd** — это расположение пустой SD-карты, введите **sudo dd if=jasperdisk-image.img of=/dev/sdd**, чтобы записать образ. Сделав это, загрузите Pi с SD-карты. *Jasper* можно настроить с удаленного компьютера через SSH, и если Pi подключен через Ethernet, вы готовы начать. Если вы используете Wi-Fi, придется присоединить к Pi монитор, запустить среду X и с помощью графической утилиты настройки Wi-Fi установить соединение.

Логин и пароль для образа *Jasper* те же самые, что и для обычного дистрибутива Raspbian, то есть pi:raspberry. Когда сеть будет настроена, соединитесь с Pi через SSH. Затем установите клиент *Jasper* в домашнюю директорию своего Pi по **git clone https://github.com/jasperproject/jasper-client.git jasper** и обновите библиотеку *setuptools* командой **sudo pip install upgrade setuptools**. После этого найдите нужные компоненты Python с помощью **sudo pip install -r jasper/client/requirements.txt**. И, наконец, настройте должные разрешения внутри домашней директории с помощью **sudo chmod 777-R *** и перезагрузитесь.

Во время перезагрузки *Jasper* запустит скрипт *boot.py* и сгенерирует файл языковой модели **languagemodel.lm** внутри папки клиента **~/jasper/**. Убедитесь, что файл создан. Если нет, скрипт *boot.py* можно вызвать вручную: **python ~/jasper/boot/boot.py**. Как только файл создастся, переходите к созданию профиля

пользователя, чтобы *Jasper* вас узнал. Перейдите в директорию клиента **~/jasper/** и запустите скрипт заполнения профиля с помощью **python populate.py**. Скрипт спросит у вас имя, контактную информацию и т. д. *Jasper* использует эту информацию, чтобы общаться с вами и отвечать на ваши запросы более точно. Помните также, что *Jasper* попросит у вас пароль вашей электронной почты и затем будет хранить его в виде простого незашифрованного текста. Подробности см. в файле **profile.yml**. Следуйте инструкциям в документации на сайте проекта для интеграции с такими сервисами, как Facebook и Spotify. Поработав со скриптом создания профиля, перезапустите Pi.

Вернувшись, *Jasper* поприветствует вас с помощью синтезатора голоса, который вы выбрали при создании профиля. Теперь вы можете приступить к взаимодействию со своим новым ассистентом, произнеся «Джаспер». Если *Jasper* вас понял, он издаст писк. Теперь можете произносить команды, например, «Который час?» или «Есть мне сообщения по электронной почте?». Если *Jasper* вас слышит, он опять-таки ответит. В зависимости от качества вашего микрофона и ораторских способностей оператора, ваши первые беседы с *Jasper* могут оказаться не самыми приятными.

Задача №10: Minecraft Pi Edition

Minecraft Pi Edition — это урезанная версия популярного *Pocket Edition*, но компонентов в ней достаточно, чтобы подстегнуть ваш творческий потенциал. Вы можете исследовать случайным образом сгенерированные миры и использовать режим строительства для создания собственных миров.

Лучшее в ней — API, доступ к которому можно получить через Python. Вы можете создать скрипты Python для перемещения игрока, а также для создания и уничтожения блоков. Это позволит вам за секунды создавать структуры, на что вручную ушли бы целые часы. Кроме того, API позволяет создавать интерактивные объекты, такие, как часы, и наделяет вас возможностями телепортации. Развлекаясь, вы осваиваете программирование.

Чтобы установит эту Pi-версию, зайдите на pi.minecraft.net и скачайте сжатый архив. Затем распакуйте его в свою домашнюю директорию с помощью **tar zxvf minecraft-pi.tar.gz**. Это поместит большой архив в директорию **~/mcpi**. Перейдите в эту директорию и запустите игру с помощью **./minecraft-pi**. Для тех, кто не знаком с *Minecraft* — вы управляете движениями мышью и клавишами WASD. Клавиши 1–8 выделяют объекты в вашей быстрой панели, пробел позволяет прыгать, а двойной удар по пробелу запускает умение летать.

Теперь вы можете использовать API для соединения с работающим экземпляром *Minecraft*. Однако прежде чем брать на себя управление с помощью API Python, неплохо будет продублировать папку Python API из папки **/mcpi** в новом месте. В терминале введите **mkdir ~/mycraft**, чтобы создать папку, и скопируйте в нее API с помощью

```
cp -r ~/mcpi/api/python/mcpi ~/mycraft/minecraft
```

Сохраним свой индивидуальный скрипт в директории **~/mycraft**. При работающей игре, нажмите клавиши Alt+Tab, чтобы снова переключиться в терминал. Откройте в терминале новую вкладку и перейдите в директорию **mycraft**. Запустите текстовый редактор, создайте новый скрипт *demo.py* с помощью **nano ~/mycraft/demo.py** и скопируйте в него следующее:

```
# Демо-скрипт взаимодействия со средой Minecraft
import minecraft.minecraft as minecraft
import minecraft.block as block
import time
# Подключимся к Minecraft, создав объект minecraft
mc = minecraft.Minecraft.create()
# Поместим сообщение в окно чата Minecraft
mc.postToChat ("Привет, это демо API Minecraft.")
time.sleep(2)
playerPos = mc.player.getPos() # Найдем позицию игрока
# Изменим позицию игрока
mc.postToChat("Телепортируемся-ка на 50 блоков в небо!")
time.sleep(2)
mc.player.setPos(playerPos.x,playerPos.y + 50,playerPos.z)
# Ждем 10 секунд, пока падаем
time.sleep(10)
#Создадим блок STONE [камень] перед собой
playerPos = mc.player.getTilePos()
mc.setBlock(playerPos.x+1, playerPos.y+1, playerPos.z, block.
STONE)
time.sleep(5)
# Превратим этот блок в WOOD [древесина]
mc.setBlock(playerPos.x+1, playerPos.y+1, playerPos.z, block.
WOOD_PLANKS)
time.sleep(5)
# Создадим башню, укладывая блоки
for top in range(0, 10):
  mc.setBlock(playerPos.x+3, playerPos.y+top, playerPos.z, block.
STONE)
time.sleep(5)
# Телепортируемся на вершину башни
mc.player.setPos(playerPos.x+1, playerPos.y+10, playerPos.z)
```



› В LXF186 Джонни продемонстрировал, как построить пушку и разнести всё в щепки в *Minecraft*.

Сохраните скрипт и, при работающем *Minecraft*, запустите скрипт командой **python ~mycraft/demo.py**. Если вы нигде не сделали ни одной опечатки, вы увидите приглашение в игру, указанное в строковой переменной **postToChat**. Потом найдем положение своего игрока по **getPos** и сохраним его в переменной **playerPos**. Положение игрока определяется координатами X, Y и Z, показанными в верхнем левом углу экрана. Затем мы используем **setPos**, чтобы изменить свое положение и телепортироваться в небо. Когда вы снова упадете вниз, мы используем параметр **setBlock** для создания каменного блока (**block.STONE**) перед вами, а через пять секунд заменим его деревянным блоком (**block.WOOD_PLANKS**). Далее используем цикл For, чтобы свалить десять каменных блоков в кучу невдалеке, потом усадим вас на самый верх — и завершим демонстрацию.

Это на самом деле всего лишь демонстрация способностей Python. Скачайте с http://bit.ly/MinecraftPyCheatSheet PDF-шпаргалку параметров API с примерами и краткими описаниями. Использование Python для управления *Minecraft* наделяет вас небывалой мощью и гибкостью, которую нам, конечно, не показать на ограниченном пространстве статьи. Но если мы вас заинтриговали, загляните в учебники Джонни по Python и *Minecraft* в предыдущих выпусках (см. Академия кодинга, **LXF185**, стр. 86, и далее). LXF


Майкл Шайлоу
ARDUINO

Умелец, учитель, художник, мастер

Майкл Шайлоу — художник и специалист по Arduino, которому нравится разбирать вещи, чтобы создавать новые. **Мэтт Хэнсон** беседует с ним о его хобби.



Майкл Шайлоу [Michael Shiloh] — мастер на все руки, сторонник свободного и открытого ПО и оборудования, преподаватель и художник. Он — активный участник DIY Maker Movement [Do It Yourself — движение, объединяющее любителей создавать вещи своими руками, — *прим. пер.*], его чертежи и роботы призваны вдохновлять на творчество; именно поэтому он широко использует платы Arduino.

Linux Format: **Во-первых, Майкл, вы занимаетесь невероятно интересными вещами, сочетая изготовление чего-либо своими руками, электронику и живопись. Как же удаётся присовокупить сюда и открытое ПО?**

Майкл Шайлоу: Хороший вопрос; я сам над этим много думаю, ведь я ещё и педагог, и в моём представлении преподавание и Open Source неразрывно связаны. Ведь Open Source — не только программы, но и знания о том, как они работают. И безусловно, те, кто желает открыто делиться своими программами, хотят одновременно и поведать миру, как они к этому пришли. Поэтому делятся они не только кодом, но и знанием.

Что в какой-то степени приближает нас к искусству. По своему опыту я сужу, что в мире искусства есть два типа людей — хотя, конечно, это только обобщение. Но, пользуясь им, одни стремятся скрывать свою работу, являя миру только видимый результат, а не то, как они к нему пришли. А другие как раз с радостью покажут вам всю кухню. И это касается

»

не только технического ремесла, которое я имею в виду, но и визуального искусства, где у каждого может быть своя техника. Возможен даже, я не знаю, вариант какого-то особого наложения красок, и один художник скажет: «Я не скажу вам, как я добился такого цвета, это мой секрет». А другой: «С удовольствием поделюсь с вами своим рецептом!» Что во многом напоминает открытое и закрытое ПО. То же самое и со зрителями. Когда я вижу произведение искусства, мне сразу хочется заглянуть внутрь. Как оно устроено? Как оно создавалось? Соответственно, есть и те, кто просто будет получать удовольствие от просмотра, не интересуясь, что внутри. И хотя, как я уже говорил, делить всех на два лагеря, по сути, неправильно, иногда полезно подумать в этом ключе.

LXF: Это точно. Как вы и сказали, есть художники, которые любят скрываться за завесой тайны, считать себя вне искусства и работать вне контекста, а есть те, что готовы «приоткрыть завесу».

МШ: Да, и обе эти позиции следует принимать: по-моему, это просто два разных подхода. Ни один не является ни лучшим, ни более правильным.

LXF: Какого рода проектами доводилось заниматься вам?

МШ: Пожалуй, их все можно разделить на две категории. Или, может, на три. Первая — это техническое искусство, которое ещё иначе называют кинетическим или инженерным. На эту стезю я ступил, сотрудничая с группой Survival Research Labs. Существует она уже более 30 лет. Это творческое объединение, где все артисты — машины. Они их сами же создают — в основном, большие и, как правило, радиоуправляемые, и ставят с ними представления. А люди выступают в качестве операторов, работающих как бы за кулисами, за кадром. Так что во многом моё инженерное искусство связано с этой деятельностью.



ПРО РОБОТОТЕХНИКУ КАК ИСКУССТВО

«Survival Research Labs — творческое объединение, где артисты — машины.»

Кроме того, я выпустил несколько коллекций мебели, довольно схожих между собой, поскольку везде использовалось много металла, много бетона. Скажем так, промышленных материалов. И в этом тоже проявляется связь с инженерией: как правило, все они концептуальны. Меня очень интересует концептуальное искусство.

Третья категория трудно поддаётся описанию. Это создание таких ситуаций, в которых люди ощутили бы значимость своего искусства. Например, был один робот, построенный довольно давно, лет 20 назад. Мне пришлось порядком покопаться в интернет-архивах, чтобы найти исходный сайт. Робот был очень прост: несколько датчиков, пара двигателей. И действовал полностью под управлением сервера Linux. Находился он



в Интернете, абсолютно в открытом доступе. Была соответствующая web-страница и связанный с ней домен http://linuxrobots.org. А робот не мог ничего делать, пока кто-нибудь не войдёт в систему и не напишет программу. Так вот, я привёл на web-странице несколько примеров и объяснил, как он работает. Всё задокументировал, написал инструкции для входа в систему. И дал пароль суперпользователя, чтобы люди со всего мира могли подключаться к нему и писать программы для этого робота. Результаты эксперимента получились очень интересными: робот весь день бездействовал, а затем вдруг просыпался и начинал носиться туда-сюда! Как правило, посреди ночи; то есть большинство моих посетителей жили в другом полушарии... а может, им просто не спалось. Многие не оставляли никаких комментариев, и я понятия не имею, кто они, откуда, чем занимаются. Некоторые оставляли очень полезные комментарии. Особенно запомнилось, как кто-то захотел научить робота танцевать — и для этого экспериментировал со всякими движениями. Мне бы такое и в голову не пришло.

То есть меня зацепила как раз идея того, что сначала людям даётся чистая доска, на которой уже они сами должны что-то написать, и с тех пор я сделал ещё ряд подобных проектов. И до сих пор в поиске...

Иногда бывает сложно доказать, что это тоже искусство. Но у меня есть такие примеры. Был у нас подобный проект, тоже с роботом, только поменьше, который мы сделали в картинной галерее. Людям предлагалось сесть и попрограммировать его с ходу. И, что удивительно, они это делали. Я всегда говорил, что показатель настоящего успеха в галерее — это способность конкурировать с бесплатным сыром и вином...

LXF: Фантастика. Интересно, а сколько посетителей обычной картинной галереи смогли бы вот так просто программировать — или же это именно ваша публика?

МШ: Не обязательно: этот проект с Arduino мы делали в самой обычной галерее, так что он был очень доступным, мы приводили примеры и объясняли, что к чему. И да, вы абсолютно правы, это не для всех. Но там были люди, до этого вообще никогда в жизни ничего не программировавшие, а попробовали — и у них получилось. Как сказал один мой друг, главное — не говорить им, что они программируют. Скажите, что они вносят изменения. Вы подсказываете им, что именно можно поменять и что будет, если они это сделают, а потом просто даёте им самим попробовать. Тогда они не думают, что программируют. И в этом, кстати, есть ещё один интересный аспект. Мне всегда было любопытно, что если при обучении технологии вы сообщаете людям, что им предстоит освоить нечто сложное, например, программирование, у них возникает фобия, или они не представляют себя программистами; так не станут и пытаться. Но если вы подадите это как-то иначе, то многие из тех, кто мог бы отказаться, попробуют. Я понял это на первой Maker Fair [Ярмарке мастеров], где мы провели огромный мастер-класс о том, как разбирать одни вещи, чтобы получались другие. Благо, пространство позволяло. Туда приходили целыми семьями, почти все садились и говорили: «Мы только посмотреть», и только у одного из них хватало решимости что-либо построить. Чаще всего это был маленький мальчик. Но чуть позже подключались и родители. Материалов была куча, так что хватало на всех. Другие дети начинали со всем этим играть, как правило, с теми, кто сидел поблизости. Они наблюдали за тем, что делают другие, и начинали помогать. Несколько часов спустя у них получались чудесные вещи, но что интересно, если бы я сказал кому-то из них «Вы должны что-то придумать», никто бы не взялся. Но благодаря тому, что им было где сесть, они

втянулись. Вот так и приходишь к проектам с роботами. Как ещё мне заманить людей, чтобы у них не возникла защитная реакция из серии: «Ой, разве я сумею»?

LXF: **Частенько многие при слове «программирование» думают: «Но у меня же нет специального образования, я не смогу». А если предложить им научить робота танцевать? Задача становится чуть ближе к ним, и они вдруг говорят: «Пожалуй, я могу попробовать!» Мы проходили это, привлекая британских детей к изучению программирования.**

МШ: Вы правы, и мне кажется, именно поэтому это очень интересно с образовательной точки зрения. При обучении робототехнике, электронике и программированию сталкиваешься с множеством студентов, кому это, так или иначе, интересно. Но не меньше будет и тех, кто постарается от этого отвертеться, несмотря на все ваши старания. Первые точно добьются успеха. Но как насчёт вторых — как до них достучаться?

ПРО ИСТОРИЮ КОМПЬЮТЕРОВ

«Об Open Source я узнал прежде, чем это стало термином...»

LXF: **В Open Source вы пришли через профессиональную деятельность? Как к логическому продолжению творчества?**

МШ: Думаю, Open Source я открыл задолго до того, как это стало термином. Я был студентом Университета Беркли, когда создавался Unix и Интернет — хотя тогда он ещё так не назывался, только был в разработке. Люди обменивались всем — абсолютно свободно, поскольку никто не видел в этом особой ценности! К примеру, какой-нибудь полезной программкой — так, одна из моих коллег хотела знать, когда ей приходит новое письмо по почте. И написала программу уведомления. Сейчас-то это умеет каждый почтовик, но тогда — нет. Так что программа оказалась очень полезной и разошлась по всему Интернету, и не думаю, чтобы коллега сама решила сделать её открытой, по-другому просто не могло быть.

Видимо, дело в том, что делиться информацией было привычнее чисто психологически, нежели скрывать её — так я и узнал, что такое Open Source. И только потом, столкнувшись с другими программами, я понял: «Оказывается, бывает и по-другому»!

LXF: **Ваша деятельность во многом связана с изготовлением вещей и быстрым созданием макетов — с Arduino вы познакомились в этой связи?**

МШ: Знаете, я затрудняюсь сказать, как именно познакомился с Arduino, но, может быть, и так. Я всегда был «самоделкиным» — не то чтобы однажды я проснулся и так решил! В этом суть работы — копаться в электронике, что-то придумывать, что-то создавать. Не припомню, когда именно я наткнулся на Arduino, просто название было на слуху, и я решил, что стоит узнать о нём побольше. Подобные продукты были и у других компаний. Тогда были те же BASIC Stamp, например. Было много компьютеров с последовательными портами. Разве что новый Apple был исключением. И все были в панике: что же теперь делать со всеми этими машинами, на которых они есть?

Поэтому я и пара моих друзей основали компанию, чтобы производить нечто вроде Arduino. У него был USB-порт; на самом деле, даже чип был тот же, что и у Arduino, FTDI. Были и порты ввода/вывода, которые можно было использовать поразному. И мы были довольно успешны, пожалуй, как и 200 других университетов по всему миру — вроде бы в университете Бата тоже было что-то подобное. Но бешеного успеха не было. Это не была такая бомба, как Arduino.

Когда, спустя несколько лет, появился Arduino, мы поняли, что в нашем устройстве уже нет потребности. И поскольку я отслеживал подобные вещи, я узнал и о нём. У всех этих плат были разные функции, разная цена, сложность, программное наполнение, и мне представляется, что успех Arduino в том, что им удалось найти золотую середину. И в плане сложности, и программ, и возможностей пользователя, и стоимости.

Ещё одним чудом стало то, что вокруг Arduino сразу же образовалось большое сообщество, а это и есть ключ к успеху. Поскольку это не только признание, но и поддержка, которой так недостаёт маленьким компаниям.

LXF: **Этому также способствовала открытость разработки и её быстрые темпы. Сейчас это огромная экосистема с сообществом энтузиастов. Но хотя открытые устройства кажутся логическим продолжением открытого ПО, их по-прежнему весьма немного. Что, по-вашему, этому препятствует?**

МШ: Думаю, препятствий два. Первое в том, что для воспроизведения открытой программы достаточно просто скопировать файл, а устройство придётся создавать заново. Второе же в том, что в открытом ПО, как в сфере бизнеса, деньги можно зарабатывать на чём угодно: работая в поддержке, создавая специальные версии этого ПО. В контексте же оборудования есть только один способ — продавать его. И давая всем возможность делать его самостоятельно, вы этого лишаетесь.

LXF: **Но как вы считаете, будет ли открытое оборудование развиваться, несмотря на эти препятствия?**

МШ: Хороший вопрос. Надеюсь, что да; думаю, что будет — ведь сейчас оно очень популярно. Но, вероятно, эти препятствия станут причиной...

Дело вот в чём. Сейчас есть невероятный интерес к тому, чтобы что-то создавать — движение DIY. И это не новшество, люди занимались этим всегда, так что сейчас мы имеем дело с коротким всплеском интереса. Думаю, это происходит циклически. Я не историк, но понимаю, что когда массовое производство стало доминировать, возникла и новая система представлений, с этим связанная, а именно, массовое потребление. А оно предполагает, что вы готовы выбрасывать вещи, чтобы купить новые. Вам хватает одного телефона, но если через шесть месяцев он устареет, вам понадобится другой. Но мастерам, умельцам это не по душе, ведь массовое производство обязано быть как можно дешевле и как можно более закрытым. Потому руки к нему приложить трудно. Этим и обусловлено возрождение интереса что-то мастерить самостоятельно, »



но я боюсь, что это всего лишь часть цикла. Что сейчас этот интерес есть, а потом он пройдёт. Думаю, то же касается и открытого оборудования, ведь, как правило, это продукты сообщества. И если для сообщества это утратит актуальность, это сразу скажется. Но есть и надежда, что это не временный всплеск, а растущий интерес — новая тенденция. Если в связи с DIY это так, то и открытое оборудование будет тоже набирать популярность.

ПРО RASPBERRY PI

«Я не фанатик Open Source, но предпочёл бы, чтобы Pi стал открытым.»

LXF: Безусловно, в использовании открытого ПО есть и этический элемент, поскольку оно доступно тем, кому не по карману дорогие лицензии на закрытые аналоги. С открытым оборудованием — как бы мы ни старались предоставить необходимые материалы и инструменты, это не всегда возможно. В своём профиле в LinkedIn вы утверждаете, что этические вопросы доступности ресурсов вас очень волнуют; является ли их решение первоочередной задачей для вас?

МШ: Не знаю, насколько я в этом преуспел. Мне всегда нравилось что-то чинить, и в этом, безусловно, есть важный этический аспект. Как и эстетический, и... ну, не знаю, мне приятно это делать. Возможно, тут есть и своя этика: я рад, что могу починить вещь или использовать заново, вместо покупки новой. И в моей личной работе мне нечего доказывать, это само собой разумеется, такова моя натура. А в ходе преподавания или подготовки к мастер-классам — и там, и там мы часто используем много переработанной и списанной электроники. Для трёхчасового мастер-класса это не совсем то, но вот недавно я подготовил двухнедельный мастер-класс для Калифорнийского госуниверситета. И раз у нас было 2 недели, мы попросили студентов заранее собрать старые приборы, и говорили о том, как их можно разобрать, что полезного из них взять, чтобы они начали разбираться в компонентах и использовать их заново. Затем мы научили собирать из них нечто новое. Так что, я думаю, демонстрируя это, делая материалы доступными, мы продвигаем эти идеи в самом чистом виде.



LXF: Интересно, что противоположность открытому Arduino составляет Raspberry Pi, поскольку несмотря на его популярность, главным нареканием является всё же наличие закрытых компонентов. Каково ваше мнение на этот счёт?

МШ: Пожалуй, у меня нет сформировавшегося мнения. Я не фанатик Open Source — да, я использую Android, а не iPhone, но ведь и в этом телефоне есть закрытые компоненты. Я предпочёл бы, чтобы Raspberry Pi стал открытым, поскольку это дало бы обучающие возможности, которых в противном случае нет. И я не воспринимаю его как противоположность; скорее, как ещё одну из множества аналогичных платформ.

LXF: Разумеется, и важнее всего их конечная цель, а именно — привлечь детей к созданию собственных проектов, сделать это доступным. Правда же, это куда важнее, чем степень закрытости кода?

МШ: Да, и целью Эбена [Eben Upton, создатель Raspberry Pi, — *прим. пер.*] было сделать компьютеры как можно доступнее; и если, добавив закрытые компоненты, ему удалось понизить их стоимость, то он своего добился.

LXF: Полностью согласен, и мы в Linux Format очень рады, что Raspberry Pi востребован в школах Великобритании. На ваш взгляд, Arduino играет аналогичную роль в США и во всём мире?

МШ: Да, и в США, и в остальном мире — во многом, в Европе. Я всё чаще встречаю его в школах и университетах. И всегда интересуюсь, откуда они узнали о нём, как они его применяют. Я понимаю, на университетском уровне, но в старшей школе — это удивительно, хотя и всегда радостно. Думаю, чаще всего, кто-нибудь из учителей, узнав о нём, решает: «Почему бы не применить его в деле».

Приятно, что его используют не только в обучении программированию или робототехнике, или чему-то ещё, для чего он был предназначен; учителя начинают видеть в нем средство для воплощения своих идей. Например, для изучения окружающей среды, для наблюдения за атмосферой и водой, и тому подобное; а один учитель физики как-то рассказал мне, что с помощью Arduino измерял скорость игрушечного автомобиля, двигающегося по наклонной плоскости, и проводил все классические физические эксперименты под его управлением. И когда я спросил его: «Чем именно он оказался полезен в учебном процессе?» — он ответил, что главное в том, что это привлекало детей. Они проявляли искренний интерес к этой машинке с мигающим огоньком. И тогда он решил: раз это помогает им лучше воспринимать материал, я должен его использовать! Это своего рода вдохновение для аудитории.



LXF: А какова, по-вашему, будет сфера применения Arduino Yún, этого классного гибрида Linux и Arduino?

МШ: Мне известно об отдельных проектах, но определённых тенденций я пока не вижу. Скорее всего, это будет что-то связанное с Интернетом, поскольку там есть и встроенный Ethernet и Wi-Fi — всё, что нужно. На самом деле, мастер-класс, который я недавно провёл на OSCON, был как раз об этом. Для всего, что связано с управлением роботами по сети, наблюдением за объектами и любыми управляющими или сенсорными приложениями... Yún идеально подойдёт.

Изначально целевой аудиторией для Arduino были не инженеры. Вопрос был в том, как заставить не-технарей включать и выключать светодиоды или управлять чем-либо кнопками. Показать им, что для этого не обязательно быть инженерами. А Yún — это уже настоящий компьютер, серьёзная инженерия, и теперь главный вопрос на моих мастер-классах — вписывается ли он в эту философию? Разработчики ПО для Arduino сделали так, что с него можно управлять тем, что происходит в части Linux, и можно запустить там программу, выполнить команду и получить ответ. Поэтому можно выйти через Linux в Интернет, скачать данные и получить обратную связь.

LXF: Как вы считаете, что ждёт Arduino в будущем?

МШ: По-моему, то есть не от имени самого Arduino, это только мои соображения, исходя из того, что я вижу и слышу — очевидно, что новые процессоры, новые контроллеры появляются постоянно, и было бы очень интересно поискать что-нибудь ещё, что можно «ардуинизировать». По мере роста их мощи более старые модели будут терять популярность, и мы должны не отставать там, где это нужно. Думаю, в этой связи очень важен Arduino Yún, указавший лёгкий путь работы в Сети; полагаю, он получит дальнейшее развитие в будущем. Очень важно для нас образование, и следовательно, поиск способов — подобных тем, о которых мы говорили — как использовать Arduino в школах. Поиск решений, как сделать обучение более доступным, предоставление необходимых материалов. Всё это вытекает из знакомых мне тенденций, и я могу долго об этом рассуждать, но наверняка знать не могу!..

LXF: Наконец, вы являетесь со-основателем Teach Me To Make – как там продвигаются дела?

МШ: На самом деле, лучше, чем мы ожидали. Изначально Teach Me To Make [Научи меня делать] появился только потому, что для уже упомянутого мастер-класса, который я делал ранее для Maker Fair, как выяснилось, нужен был сайт и URL к нему. Так что к Teach Me To Make я пришёл совершенно случайно: просто создал домен, чтобы разместить там материалы, картинки и прочее. Как организация, Teach Me To Make появилась, когда мой партнёр Джуди и я стали проводить мастер-классы вместе и материалы понадобилось куда-то выкладывать. Также нужно было рекламировать то, что мы делаем, и о любом мастер-классе, любом мероприятии, любом новом проекте мы сообщали на сайте. Там есть блог на *WordPress*, потому что это просто, и мне нравится, когда можно редактировать по ходу, а *WordPress* как раз в этом силён. Я туда выложил кучу материалов — главным образом, на основе вопросов, которые мне задавали студенты.

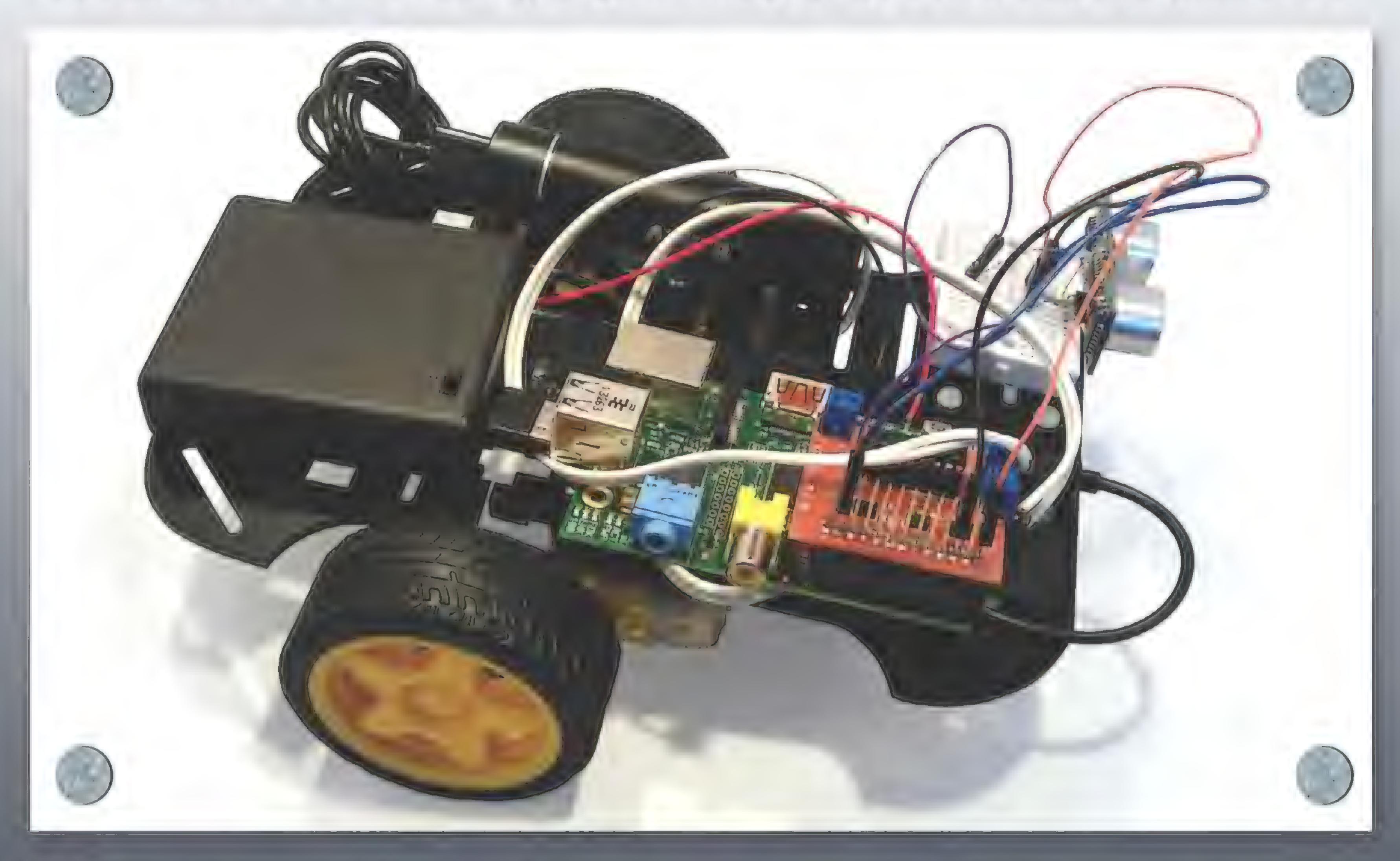
ПРО TEACH ME TO MAKE

«Мы не заботимся о раскрутке, но нас посещают, спрашивают совета.»

Одним из самых популярных вопросов был «Можно ли использовать транзистор в качестве коммутатора?» Поэтому я там подробно его разбираю. Удивительно, но это востребовано, вопросы поступают со всех концов земли. Следовательно, несмотря на то, что мы не особенно заботимся о раскрутке Teach Me To Make, люди его посещают, связываются с нами, спрашивают нашего совета. LXF

Raspberry Pi: РОБОТЫ

Лес Паундер раскурочивает Raspberry Pi, открывая новую, абсолютно безопасную для людей цивилизацию роботов.



Роботы были центральной темой научной фантастики, начиная с 1950-х: их серебристые фигуры маячили над человечеством, норовя уничтожить его сотней разных способов. На самом же деле, роботы честно вкалывали с нами на наших заводах, в море и в космосе. То есть везде, где человеку это было бы чревато опасностью для жизни.

Но сегодня роботы — уже не просто дорогостоящий инструмент для заводов и предмет роскоши для богачей. Во многом благодаря дешёвым платформам разработки, таким как Raspberry Pi, мы можем использовать роботов в нашей повседневной жизни и, что ещё важнее, в наших школах. Создание роботов — занятие увлекательное и полезное, позволяющее в одном проекте упражнять многие навыки: в сборке, программировании и электронике.

> «Создание роботов позволяет в одном проекте упражнять многие навыки.»

Для программирования робота применяется Python — самый распространённый язык, которому обучают детей в британских школах.

А силами Raspberry Pi Foundation, ещё и главный язык проектов на основе Pi. Чтобы построить робота, нужно приобрести комплект деталей в каком-нибудь интернет-магазине, что мы и сделали для нашей статьи. Но можно и придумать своего собственного, благо компоненты на рынке представлены в изобилии.

Электроника робота — это различные соединения, подключаемые к аппаратной части посредством GPIO-контактов (General Purpose Input/Output) Raspberry Pi. Последний в этом случае взаимодействует с платой управления электродвигателем и ультразвуковым датчиком. При подключении к Raspberry Pi эти компоненты становятся расширением нашего Pi, позволяя окунуться в мир физических расчетов.

И детям, и взрослым для успешной работы необходимо, чтобы проекты были интересными; именно в этом и пересекаются роботы, Python и Raspberry Pi. Используя Raspberry Pi и несколько экономичных компонентов, можно построить простого робота, который будет пытаться найти выход из лабиринта, написав всего 80 строк кода Python. К концу этой статьи ваш первый робот будет готов познавать мир.

Плата управления двигателем Ryanteck фантастически проста в использовании, но сначала её нужно спаять. И если в этом деле вы новичок, рекомендуем приобрести её в уже готовом виде: это обойдётся всего на несколько фунтов больше. Ryanteck продает плату управления двигателя как часть комплекта для робота, и это наиболее экономически выгодный способ подобрать все необходимые детали.

Плата относительно проста в сборке, и Ryan предоставляет отличные инструкции на http://ryanteck.uk/rtk-000-001-assembly, следовать которым очень легко. Равно как и разобраться с прилагаемой монтажной панелью — инструкции для нее также есть на сайте, однако не стесняйтесь

3.3V	1	2	5V
I2C1 SDA	3	4	5V
I2C1 SCL	5	6	GROUND
GPIO4	7	8	UART TXD
GROUND	9	10	UART RXD
GPIO 17	11	12	GPIO 18
GPIO 27	13	14	GROUND
GPIO 22	15	16	GPIO 23
3.3V	17	18	GPIO 24
GPIO 10 MOSI	19	20	GROUND
GPIO 9 MISO	21	22	GPIO 25
GPIO 11 SCLK	23	24	GPIO 8
GROUND	25	26	GPIO 7

› В GPIO есть две схемы контактов: логическая, под названием 'Board', и 'Broadcom BCM'. Мы описываем вторую.



› Ультразвуковые датчики, которые служат «глазами» нашего робота, подобны парковочным датчикам современных автомобилей.

адаптировать всё под свои нужды. При добавлении таких компонентов, как аккумулятор и блок питания USB, советуем использовать Blu-tack (или любой другой временный клей), чтобы закрепить их надёжнее.

Код Python

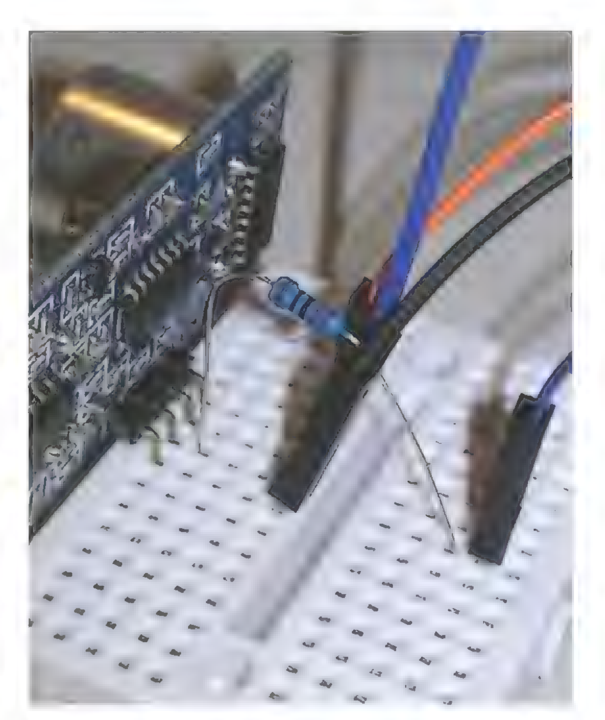
Для управления оборудованием, составляющим физическую сторону нашего проекта, требуется программа на Python, определяющая очередность шагов для нашего робота. Этот простой робот сможет двигаться вперед, назад, поворачивать налево и направо, подчиняясь командам, организованным в последовательность.

Чтобы запустить последовательность, мы воспользуемся ультразвуковым датчиком, который служит нашему роботу органом зрения. Употребив псевдокод, помогающий в доступной форме объяснить принцип действия программ, давайте рассмотрим эту последовательность.

```
Импортировать все необходимые модули
Создать все переменные
Настроить GPIO Raspberry Pi
Создать функцию для управления датчиком ульттразвука
Создать функцию для управления двигателем
Создать бесконечный цикл
Вызвать ультразвуковую функцию, чтобы измерить расстояние до объекта
Создать условный оператор
Если расстояние от робота до объекта больше, чем 10 см, идти вперед
Или Если расстояние от робота до объекта меньше, чем 10 см, повернуть направо
```

Итак, пройдёмся по нашему коду, и в первую очередь рассмотрим модули.

Модули — это внешние библиотеки Python, и их легко импортировать в ваш проект. Например, через модуль Rpi.GPIO мы задействуем контакты GPIO нашего Raspberry Pi. Импортировать модуль, например, модуль времени, мы можем просто командой **import**. В нашем коде мы делаем



› Резисторы с виду невелики, но играют важную роль в защите вашего Pi от перегрузок.

это дважды: один раз — для модуля времени, ещё один — для функциональности GPIO.

```
import RPi.GPIO as GPIO
import time
```

Как видите, модуль времени импортируется иначе, чем модуль RPi.GPIO. Используя метод 'as', мы можем переименовать модуль, чтобы с ним было легче работать, сократив RPi.GPIO до GPIO. При дальнейшей работе с кодом это нам пригодится.

Далее мы видим три переменных, одна из которых является особенной — это глобальная переменная **distance** [*англ.* расстояние]. Глобальные переменные — это особый тип переменных, которые используются как внутри, так и вне функций (мы рассмотрим их позже в данном проекте). Ещё две переменные — **echo** и **trigger** — содержат номера контактов GPIO, используемые датчиком ультразвука.

```
global distance
```

»

```
trigger = 11
echo = 8
```

Теперь обратимся к настройке контактов GPIO. Во всех моделях Raspberry Pi есть две схемы расположения контактов GPIO: одна логическая система нумерации контактов называется BOARD, другая — BCM. BCM — это аббревиатура Broadcom, создателей System-on-a-Chip (SoaC, однокристальная система или система-на-одномчипе), используемой в Raspberry Pi. Схема BCM не является логической; скорее, она отбирает необходимые пути для подключения SoaC к GPIO, позволяя нам их задействовать.

В этом проекте мы будем использовать схему BCM, как указано в документации Ryanteck.

```
GPIO.setmode (GPIO.BCM)
```

Независимо от того, BOARD у вас или BCM, вы можете использовать GPIO в ваших проектах, но сначала надо определить, за что будет отвечать каждый контакт. Например:

```
GPIO.setup (17, GPIO.OUT)
```

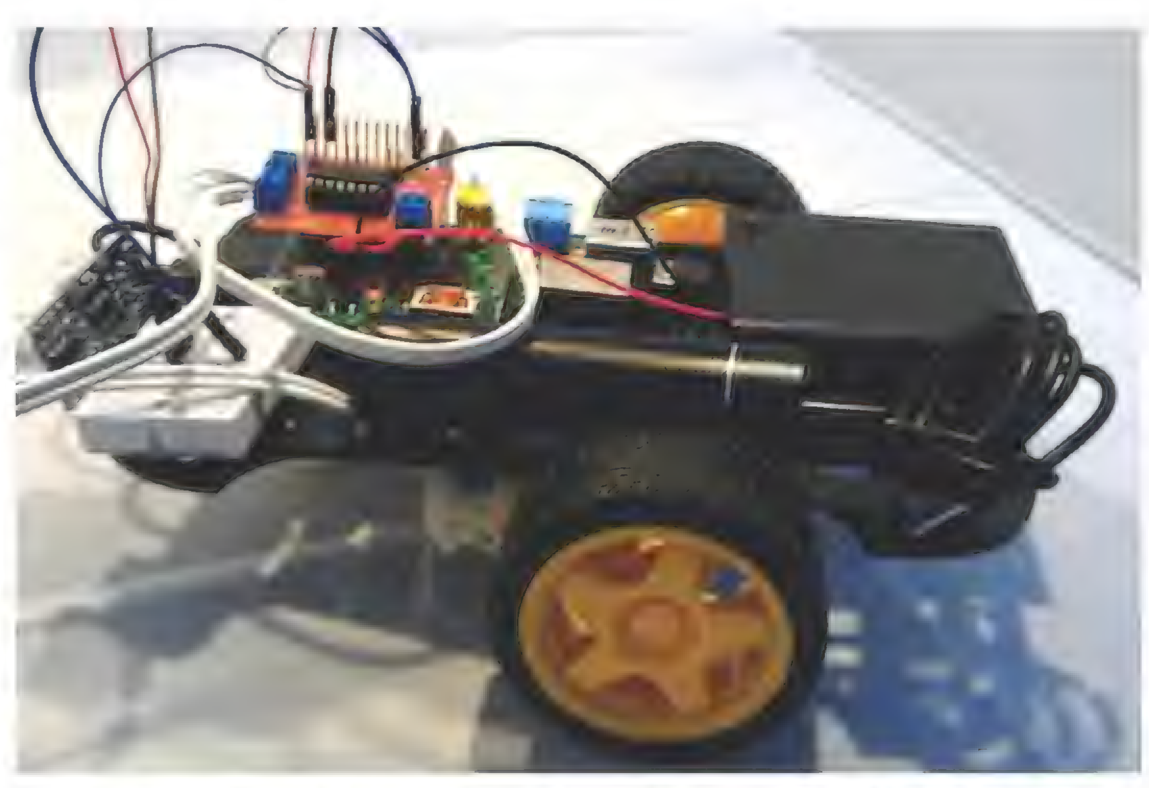
Это сообщает нашему проекту, что контакт 17 настроен как выход, следовательно, по нему будет передаваться ток к плате управления двигателем Ryanteck.

Вместо указания конкретных чисел можно также использовать переменные, содержащие номера контактов, что мы и сделаем в нашем проекте с помощью **echo** и **trigger**.

```
GPIO.setup(echo, GPIO.IN)
GPIO.setup(trigger, GPIO.OUT)
GPIO.setup(17, GPIO.OUT)
GPIO.setup(18, GPIO.OUT)
GPIO.setup(22, GPIO.OUT)
GPIO.setup(23, GPIO.OUT)
```

Измерения и их результаты

Плата Ryanteck использует четыре контакта GPIO для управления двигателями на нижней стороне нашей монтажной панели. Это контакты 17, 18, 22 и 23. Обычно 17 и 18 связаны с левым двигателем, а 22 и 23 — с правым. Левый двигатель подключен к M1 на плате, а правый — к M2.



› Где мы окажемся? Связано ли будущее Ubuntu, как и компьютерных технологий в целом, с мобильными устройствами?

Завершив сборку, мы можем создать ряд функций, содержащих весь код, необходимый для выполнения конкретных задач. Первая функция касается нашего ультразвукового дальномера.

Ультразвуковые датчики посылают ультразвуковой импульс к объекту, а затем ожидают, когда он вернётся обратно. Время отклика используется для расчёта расстояния между датчиком и объектом. В нашем коде мы сначала определим имя функции; учтите, что ей потребуется аргумент, в данном случае **sensor**, — эта переменная будет служить обозначением датчика, что в дальнейших проектах позволит использовать несколько таковых. Также мы задействуем переменную 'global distance', созданную ранее.

```
def ultra(sensor):
  global distance
```

Теперь составим условное выражение, в данном случае, **if...else**. Начнем с «если аргумент датчика равен нулю, ждать 0,3 секунды», чтобы дать ультразвуковому датчику время «прийти в себя». Далее мы велим контакту запуска подать ток на ультразвуковой датчик на 0,00001 секунды, этого хватит для короткого импульса. Затем выключим контакт запуска, остановив импульс.

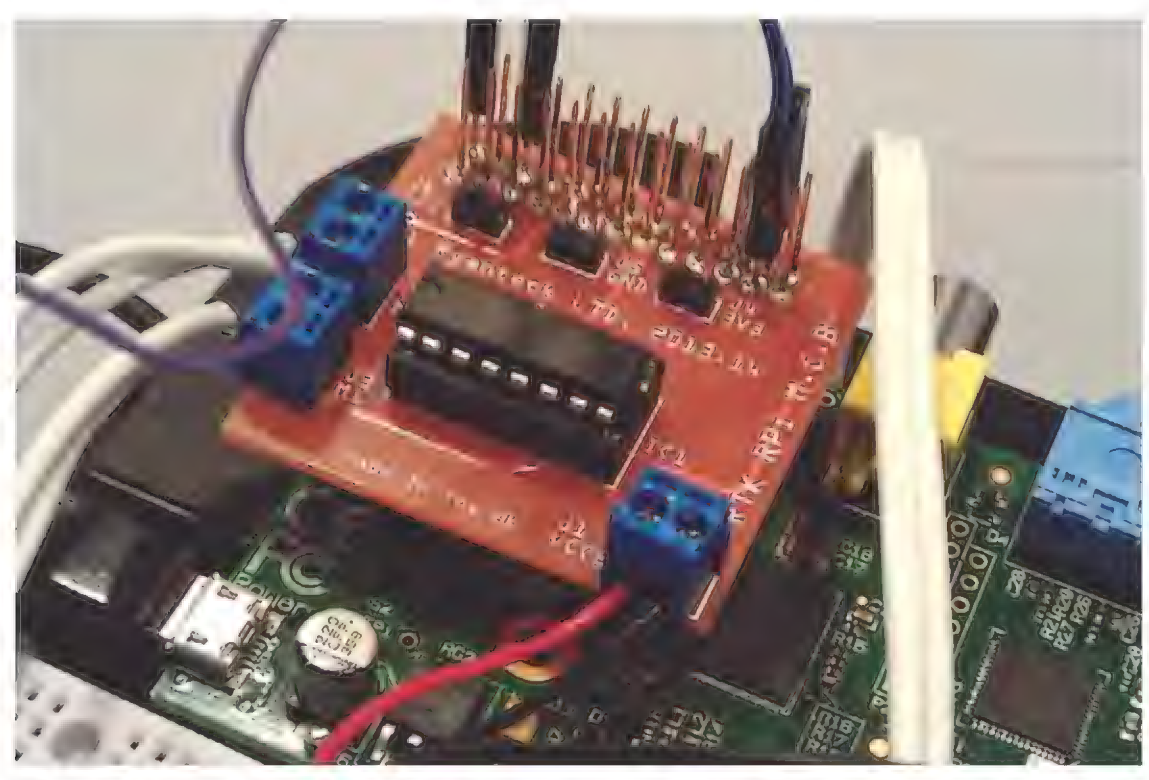
```
  if sensor == 0:
    time.sleep(0.3)
    GPIO.output(trigger, True)
    time.sleep(0.00001)
    GPIO.output(trigger, False)
```

После отправки импульса датчик дожидается получения эхо-сигнала, а в это время код использует следующую логику: пока сигнала нет, переменная **signaloff** функцией **time.time()** устанавливается в текущее время. Когда отклик прибудет, в переменной **signalon** сохранится время его получения.

```
    while GPIO.input(echo) == 0:
      signaloff = time.time()
    while GPIO.input(echo) == 1:
      signalon = time.time()
```

Теперь обратимся к школьной математике, чтобы вычислить расстояние до объекта. Создадим переменную с именем **timepassed** и запишем в ней разность значений **signalon** и **signaloff**. Получив этот ответ, мы создаём еще одну переменную, под названием **distance** — это наша глобальная переменная, которая хранит значение **timepassed**, умноженное на 17000. Зачем умножать на 17000? 34000 см/с — это скорость звука, а за время **timepassed** наш импульс проходит расстояние до объекта дважды (туда и обратно), поэтому мы заранее разделили 34000 пополам. Наконец, мы получаем расстояние в выводе переменной **distance**.

```
    timepassed = signalon - signaloff
```



› В отличие от других, плата Ryanteck не требует специальных модулей Python и потому идеально подходит для небольших проектов.

```
    distance = timepassed * 17000
    return distance
```

Наконец, мы завершаем выражение **if** и переходим к части **else**. **Else** выполняется, если первое условие ложно. В этом случае просто отобразим на экране 'Ошибка', и процесс будет выполнен заново.

```
  else:
    print "Ошибка."
```

Другие функции управляют двигателями, подключёнными к плате управления.

Собираем всё вместе

Чтобы робот проехал вперед, необходимо, чтобы оба двигателя работали вместе и в одном направлении, а при настройке мы назначили для них контакты 17 и 23. Чтобы включить их, мы используем число 1, которое в логике программирования обозначает True [Верно] либо Высокий уровень сигнала. Затем подождём секунду, чтобы робот успел продвинуться на несколько сантиметров. Далее, отключим контакты 17 и 23, используя 0, что означает False [Ложный] либо Низкий уровень сигнала.

```
def forward():
  GPIO.output(17,1)
  GPIO.output(23,1)
  time.sleep(1)
  GPIO.output(17,0)
  GPIO.output(23,0)
```

Данную функцию можно применить и для управления поворотами робота. Нужно будет лишь немного подправить её, чтобы робот мог поворачиваться на месте, как танк. Вот код для поворота налево:

```
def left():
  GPIO.output(17,0)
  GPIO.output(18,1)
  GPIO.output(22,0)
  GPIO.output(23,1)
  time.sleep(1)
  GPIO.output(17,0)
  GPIO.output(18,0)
  GPIO.output(22,0)
  GPIO.output(23,0)
```

Найти нужную комбинацию для включения и выключения обоих двигателей может быть непросто, так как они умеют работать в двух направлениях. Для достижения наилучших результатов придётся повозиться.

> «Можно добавить звуковые датчики, научив робота реагировать и убегать.»

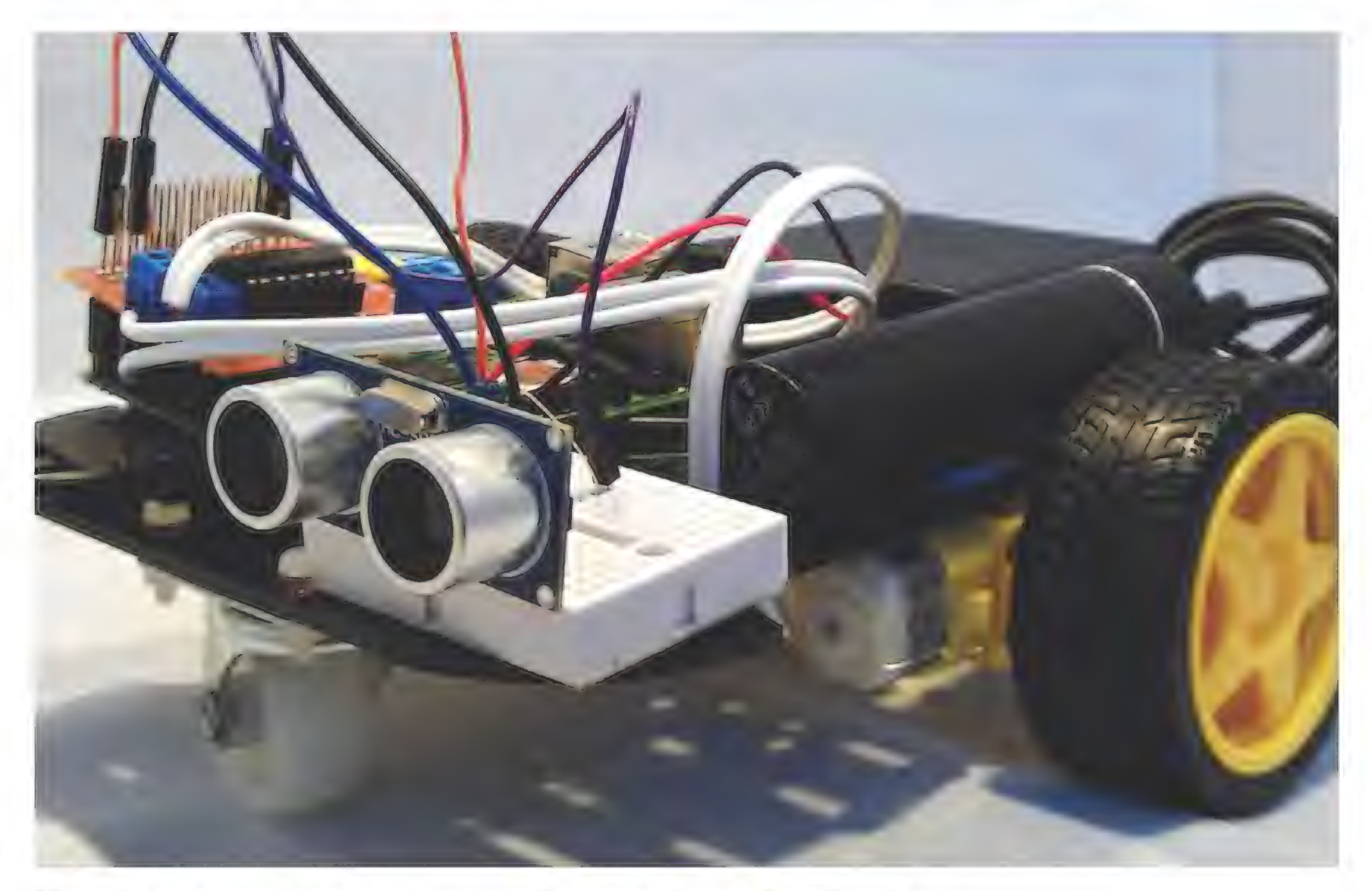
Последняя часть нашего кода проста, но эффективна. Она начинается бесконечным циклом, где вызывается ранее созданная функция измерения расстояния от робота до препятствия. Если расстояние между роботом и объектом больше 10 см, робот воспользуется функцией **forward**, чтобы двигаться вперед; если оно меньше 10 см, робот будет пытаться свернуть налево, чтобы найти выход.

```
while True:
  ultra(0)
  if distance > 10:
    forward()
  elif distance < 10:
    left()
```

Теперь наш робот готов к поиску выхода из лабиринта; ну, а как ещё может применяться этот проект? Как говаривал Рокки Бальбоа [Rocky Balboa]: «Если вы готовы на всё, чтобы добиться желаемого, кто же сможет вас остановить?»

К этой простой основе можно добавить и другие датчики, например, линейные, которые позволят нашему роботу следовать по линии, нанесённой на полу, что широко используется в промышленности. Можно добавить звуковые датчики, научив робота реагировать на звук и убегать. Можно передавать от него потоковое видео в Интернет в прямом эфире и дистанционно управлять им с обычной web-страницы.

Робототехника — прекрасная основа, чтобы познать заоблачные возможности — в буквальном смысле: построить на базе Pi беспилотный летательный аппарат-дрон. А пока наслаждайтесь своим новым роботом — можете даже назвать его «малыш Арни». LXF



› У нашего робота два колеса по бокам, которые обеспечивают перемещение, и поворотный ролик спереди.



Включите, настройте... и в JACK

Сорвем завесу тайны с JACK! **libremusicproduction.com** открывает, как подключиться к миру профессионального аудио в Linux.

Другие профессиональные руководства, советы и ответы на вопросы по музыке см. на **libremusicproduction.com**

JACK — это фактический стандарт профессионального сервера аудио для работы в Linux. JACK (рекурсивный акроним от JACK Audio Connection kit) — очень мощное программное обеспечение. Некоторые новые пользователи поначалу считают его запутанным, что и неудивительно, при его сложном интерфейсе и бесчисленных настройках. И тем не менее, нужно лишь разобраться в основах, чтобы овладеть его могуществом.

Надеемся, что к концу нашего руководства вы обретете достаточное понимание JACK и его функциональности, и сможете единожды настроить его так, чтобы потом о нём забыть и лишь полагаться на его мощь в своей работе. На протяжении всей статьи мы будем называть разные программы, поддерживающие JACK, JACK-совместимыми.

Что такое JACK

JACK — это сервер аудио, оптимизированный под требования работ по созданию музыки. Есть несколько основных аспектов его деятельности. Давайте мы их вкратце разберем.

» **Настройки:** JACK управляет вашими аудио- и MIDI-настройками. Это позволяет выбирать аудиоинтерфейс и все необходимые параметры аудио, такие как частота сэмплирования (выборки), размер буфера и периоды. Все эти настройки мы подробнее рассмотрим ниже.

» **Производительность:** Применение JACK позволяет достичь очень малой латентности (задержки) при работе с аудио и MIDI. Это значит, что при записи музыкального инструмента на компьютере можно контролировать звук через динамики или наушники без ощутимого рассогласования. Поищите 'Latency and Latency-Compensation' на bit.ly/LXFardour, там есть подробный обзор и наглядное описание латентности.

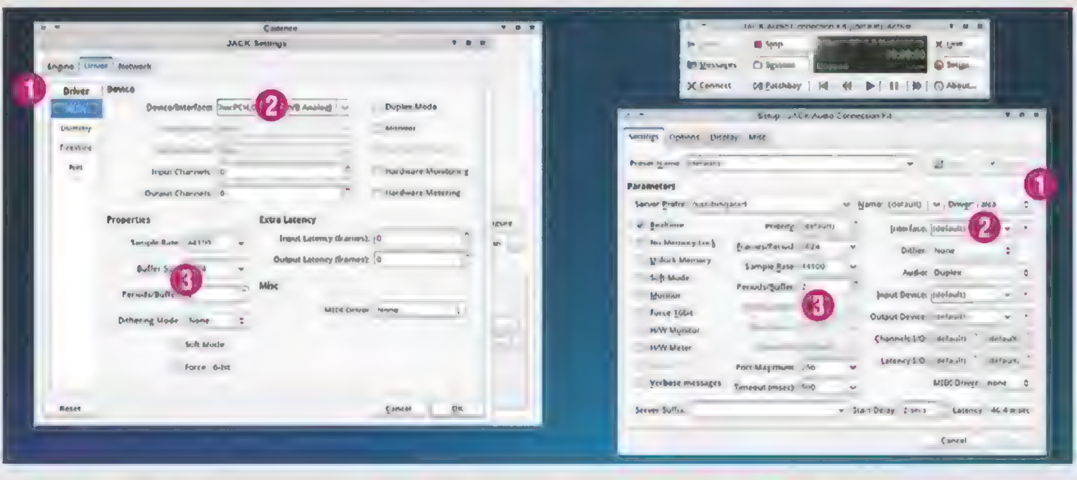
» **Подключения и взаимодействие:** Это сильная сторона JACK. Все входы и выходы вашего аудиоинтерфейса и/или JACK-совместимых программ можно коммутировать произвольно. JACK справляется с соединениями не только между программами, но и внутри самих программ. JACK-совместимые программы управляют через JACK подключениями своих входов и выходов. Прелесть тут заключается в том, что все эти связи доступны для любой другой JACK-совместимой программы. Они не ограничены тем, что являются только «внутренними».

Всё это делает JACK очень гибким инструментом и позволяет соединить любое количество JACK-совместимых программ друг с другом.

Синхронное плавание

JACK может быть использован для синхронизации нескольких программ. Это значит, что если вы хотите использовать MIDI-секвенсор от одной программы и аудиосеквенсор от другой, вы с лёгкостью синхронизируете их с помощью JACK. Обе программы запустятся и остановятся одновременно. Движение по временной шкале в одной программе будет зеркально отражено в другой программе, если та настроена на синхронизацию с JACK.

JACK, я тебя знаю



1 Выберите драйвер
Убедитесь, что вы выбрали вкладку драйвера *Cadence* (левое окно) и выберите ваш драйвер, обычно это ALSA.

2 Выберите интерфейс
Удостоверьтесь, что ваш интерфейс подключен и запущен, иначе вы можете его просто не увидеть.

3 Выберите настройки
Установите свои настройки интерфейса здесь. Если вам интересно, то кадры/период точно такие же, как в *Buffer*.

Как видите, JACK — очень умелый инструмент; но не пугайтесь его потенциальной сложности. Работа с JACK не обязана быть трудной. Его способность к мудрёным рабочим процессам вовсе не означает непригодности для более простых вариантов. Он будет делать то, что вам от него нужно, и достаточно гибок, чтобы позволить вам замахнуться на большее, если понадобится. Вы вообще можете преспокойно работать в одной программе и никогда не выходить из неё, разрешив JACK управлять всеми её связями.

Теперь мы знаем, что такое JACK и чем он хорош. Как бы нам им попользоваться? Первое, что стоит отметить: JACK сам по себе — это программа командной строки; однако для JACK есть различные графические менеджеры, которые позволяют легко раскрыть его мощь. Менеджеры JACK можно разделить на два основных типа (а вы выбирайте тот, что больше подходит вашему рабочему процессу): менеджеры настройки, позволяющие запустить JACK с конкретными параметрами, и менеджеры подключений, которые в первую очередь занимаются созданием связей.

Различные вариации менеджеров JACK совмещают два этих аспекта в разной степени. Почему же вы должны предпочесть один другим? Если вы делаете всю работу внутри DAW [Digital Audio Workstation — рабочая станция для цифрового аудио] и не используете сторонних программ, вам будет достаточно менеджера настройки JACK, чтобы просто запустить JACK с выбранными вами параметрами. В подобном случае все ваши соединения делаются в вашей DAW, так что во внешнем менеджере подключений нет необходимости.

Для случая более модульной структуры звукозаписи, содержащей несколько процессоров эффектов и программы для записи и секвенсирования, менеджер JACK, более ориентированный на создание связей, будет лучшим выбором, тем более что некоторые программы-модули не имеют собственных менеджеров подключений.

Вот вам список самых популярных менеджеров JACK:

» **Qjackctl** Очень мощный и популярный менеджер JACK. *Qjackctl* позволяет получить доступ к большому количеству настроек JACK и включает менеджер подключений, управление передачей и даже менеджер управления сессией JACK. Он появляется в небольшом окне, позволяя работать только с необходимыми для доступа настройками. И его можно свернуть на панель задач, так что он не будет путаться у вас под ногами, когда вы его установите.

» **Cadence** Простой в использовании инструмент для настройки и запуска JACK. Он поставляется как часть дистрибутива KXStudio и включает мосты JACK, позволяющие воспроизводить через JACK обычные звуки настольного компьютера, такие как флэш-видео. *Cadence* также может запускать JACK после входа в систему, включая »

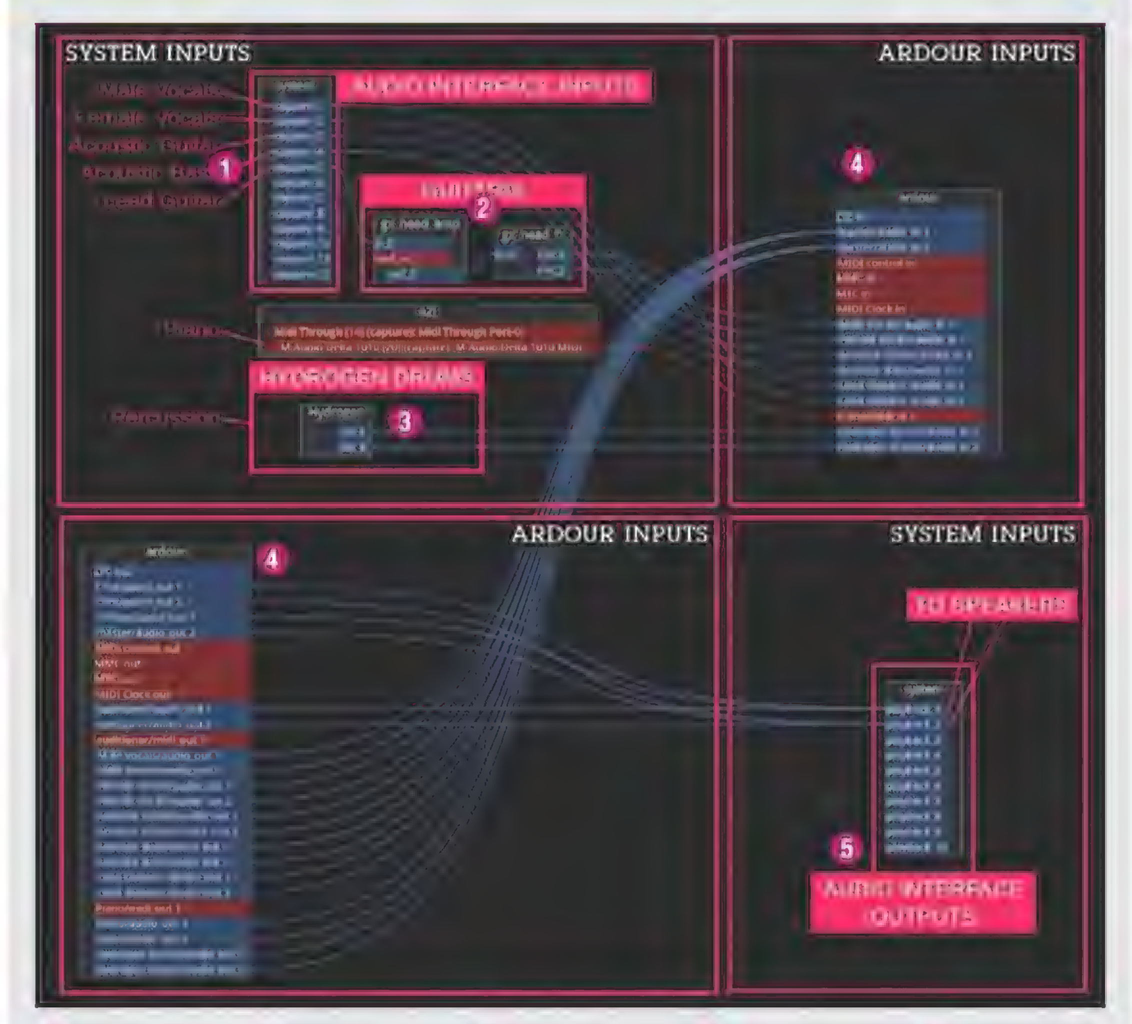
JACK-совместимость

Любое ПО, поддерживающее JACK, именуется JACK-совместимым. Это значит, что все аудио- и MIDI-входы/выходы управляются через JACK. Идея применения в DAW сторонней системы управления связями вначале немного озадачивает. Можно думать об этом, проводя параллель. Если вы привыкли к DAW, управляющей всеми вашими соединениями, считайте, что JACK — это то же самое, только в пределах всей системы. Любое JACK-совместимое ПО можно добавить в эту систему так же, как DAW добавляла бы плагины. Отличие состоит в том, что JACK даёт вам точный контроль над связями всего со всем, что к нему подключено.

Монолитность

Монолитные структуры — это когда вся работа выполняется в одной программе. Это наиболее общий подход, привычный людям, использующим ПО аудио для Windows и Mac. В Linux JACK позволяет использовать модульную настройку, хотя некоторые приложения являются полнофункциональными и могут применяться как монолитные — например, *Ardour* и *Qtractor*.

JACK в Patchage



1 Входы аудиоинтерфейса Основные входы для треков перечислены здесь.

2 GuitarIX Виртуальный усилитель гитары; работает с сигналом точно так же, как и настоящий.

3 Барабаны Hydrogen Входы драм-машины GNU/Linux для *Ardour*.

4 Ardour Самое важное: редактирование, запись и элемент микширования.

5 Выходы аудиоинтерфейса Здесь микшируется вводимая информация и отправляется на выход.

мосты JACK. Если вы один раз настроите его, особого пригляда за ним не потребуется.

» Patchage и Catia Два похожих визуальных менеджера подключений для JACK. Они дают обзор всех ваших соединений. Аудио- и MIDI-порты обозначены цветами, что упрощает определение типа соединения. Создать соединение не сложнее, чем щелкнуть мышью и перетащить с порта выхода на порт входа.

Базовые настройки

Если вы знакомы с созданием аудио на других платформах, вам, наверное, знакома и терминология по части настроек. Если вы новичок, то можете запутаться, поскольку настроек у JACK очень много. Основное, что следует помнить — то, что большая часть этих настроек не столь важна, по крайней мере, для запуска. Если вы не знаете, что делает та или иная настройка, возможно, она просто вам пока не нужна. Вот объяснение основных настроек, о которых стоит побеспокоиться, чтобы подготовиться и приступить к работе...

» Размер буфера (кадры/период): Меньшие размеры буфера создают меньшую латентность мониторинга. Более низкий уровень настроек увеличит быстродействие вашего компьютера за счет повышенного использования CPU. Более высокий уровень настроек стабильнее, но не даст такой низкой латентности при мониторинге. Если вы стремитесь к более своевременному мониторингу, настройки где-то между 64–256 должны дать вам наиболее приемлемый результат.

» Частота сэмплирования: Более высокая частота приведёт к меньшей латентности при том же уровне настроек буфера. Этот параметр зависит от оптимальных настроек вашего аудиоинтерфейса и ваших личных предпочтений. Одним нравится записывать с более высокой частотой, другим хватает и 44 100 (44,1 кГц), т. е. CD-качества. Распространена также частота 48 000 — некоторые интерфейсы лучше работают с ней. Чем бóльшую частоту вы установите, тем меньше будет латентность, но в то же время вы перегружаете CPU, что иногда приводит к щелчкам и потрескиванию [xruns].

» Периоды/буфер: для USB-устройств можно достичь более стабильного низкого уровня задержек, установив этот параметр в 3. В прочих случаях задайте его как 2. Если вы стремитесь к низкой латентности, придется искать оптимальное соотношение, не создающее перегрузок CPU. Если вам не нужно контролировать свое аудио с компьютера при низкой латентности, незачем нагружать ваш компьютер сверх необходимого. Лучше оставить высокие значения настроек, поставив в приоритет надёжность. Если у вас есть возможность аппаратого мониторинга через ваш интерфейс, лучше использовать её.

Настройка JACK

Далее будем предполагать, что у вас есть структура реального времени с правильными настройками. KXStudio (http://kxstudio.sourceforge.net) и AVLinux (http://www.bandshed.net/AVLinux.html) — дистрибутивы Linux, предлагающие подобную среду по умолчанию. Мы увидим, как настроить JACK с *Qjackctl* и *Cadence*. Следующие шаги подходят для обоих — исключений не замечено.

Шаг 1: Нажмите кнопку Setup в *Qjackctl* или кнопку configure в *Cadence*.

Шаг 2: Убедитесь, что включена опция реального времени (вкладка engine в *Cadence*).

Шаг 3: Выберите ваш драйвер аудио (в *Cadence* для этого убедитесь, что выбрана вкладка driver).



› Чтобы синхронизировать программы, вам необходимо убедиться, что одна – master [ведущая], а другая/другие – slave [ведомые].



› Встроенная в *Ardour* матрица подключений дает простой способ управлять множеством связей.

Если вы используете устройство firewire, выберите Firewire. В любом другом случае оставьте ALSA.

Шаг 4: Выберите свой интерфейс в выпадающем меню интерфейса в меню внизу.

Шаг 5: Выберите настройки для своего интерфейса.

Шаг 6: Нажмите OK, чтобы принять настройки.

Шаг 7: Жмите на Start!

В *Qjackctl*, не трогайте поле имени. Некоторые новички любят давать своему JACK-серверу имя, но это приводит к некорректной работе JACK. Оставьте такое до того момента, когда будете уверены в своих действиях, иначе вы наживёте больше проблем, чем оно того стоит.

Задав эти настройки, вы сможете использовать любые JACK-совместимые программы, и они будут запускаться с вашими настройками. Некоторые программы позволяют менять размер буфера, регулируя тем самым латентность аудио, без перезапуска JACK.

Итак, хоть JACK и очень мощный, это не должно вас пугать. По сути, настроить ваш интерфейс и запустить его очень просто. Как и с любой приличной программой, вы глубже изучите JACK по мере его использования, но не берите в голову слишком много, если всё, чего вы на самом деле хотите, это просто научиться пользоваться новым ПО.

Если у вас есть какие-либо проблемы с запуском JACK или распознаванием вашего интерфейса, просто убедитесь, что ваш интерфейс подключен и включён до открытия вашего менеджера JACK. Если проблемы не исчезли, попробуйте перезапустить систему при подключённом интерфейсе.

JACK Sync

Это ещё один полезный аспект JACK. Вы можете не только соединять программы между собой, но и быть уверены, что они работают совершенно синхронно. Мы сделали небольшой обзор того, как добиться этого, посредством демонстрации, представленной ниже.

Шаг 1: Первым делом запустите JACK через выбранный по вашему вкусу менеджер JACK.

Шаг 2: Затем запустите программы, которые вы хотите синхронизировать, в данном случае — *Ardour* и *Hydrogen*. Для начала позаботьтесь о правильной настройке этих программ.

Шаг 3: По умолчанию *Ardour* установлен как времязадающее ПО [timemaster] для JACK — нам это и нужно. Данную настройку можно найти, переместившись в Session > Properties и перейдя во вкладку timecode. Оставим её как есть.

Шаг 4: Теперь щёлкнем по кнопке Internal на панели инструментов *Ardour*. Это изменит настройки в JACK, назначив *Ardour* как timemaster для JACK.

Инструкция для *Hydrogen*

Шаг 1: Перейдите в меню Tools > Preferences. В качестве аудиодрайвера во вкладке Audio System выберите JACK и перезапустите *Hydrogen*.

Шаг 2: Теперь вы увидите две кнопки timemaster наверху на панели инструментов. Убедитесь, что J.trans (подача Jack) включено. Это означает, что он будет следовать протоколу передачи JACK, которым теперь управляет *Ardour*.

Шаг 3: Теперь обе программы будут синхронно выполнять свои протоколы передачи. Вдобавок, так как *Ardour* у нас timemaster, смена темпа в *Ardour* отразится в *Hydrogen* или любой другой подчинённой программе.

Тот факт, что JACK управляет аудио- и MIDI-портами, а также позволяет синхронизировать программы, их использующие, делает его очень мощным инструментом и позволяет создавать с его помощью модульные структуры. Тем, кто в таковых заинтересован, рекомендуем заглянуть в управление сессией, особенно в *Non Session Manager*, чтобы изучить возможности управлять модульными структурами через эти инструменты.

Создание соединений

Давайте создадим сценарий записи для демонстрации того, как создать соединения с помощью JACK. Вот сценарий применения *Ardour* для записи следующих инструментов:

ИНСТРУМЕНТЫ И ВХОДЫ	
Мужской вокал	Вход 1
Женский вокал	Вход 2
Акустическая гитара	Вход 3
Акустическая бас-гитара	Вход 4
Лидер-гитара	Запускается через виртуальный гитарный усилитель *Guitarix*
Перкуссия	Синхронизированная драм-машина *Hydrogen*
Пианино	MIDI-клавиатура, подключённая к MIDI-входу с виртуальным плагином пианино в дорожке

Для создания этих соединений использован менеджер подключений, встроенный в *Ardour*, но логика останется той же, какой бы менеджер вы ни взяли. Вверху слева вы увидите окно настроек микшера в *Ardour*. Кроме того, у нас запущены одновременно *Guitarix* и *Hydrogen*. Обратите внимание на выделение строк. Все соединения готовы, включая соединения с *Guitarix* и *Hydrogen*.

Чтобы создать эти подключения в *Ardour*, нажмите левой кнопкой мыши на кнопки входа, показанные в выделенной строке. Появится окно, где можно выбрать ваш вход. В данном случае мы хотели бы направить второй вход из аудиоинтерфейса во вторую дорожку (женский вокал), так что выберите capture 2, отметив это поле галочкой. Повторяйте эти шаги для каждой дорожки, пока все соединения не будут настроены как надо.

Стоит отметить, что все эти соединения появятся сразу во всех JACK-совместимых программах: например, посмотрев в окно соединений в *Guitarix*, вы тоже увидите соединения, созданные в *Ardour*. А дело тут в том, что эти программы соединений не создают: для этого они используют JACK, говоря ему, что делать. Помните: JACK — ваша система коммутации, а эти программы лишь используют её. Ниже находится полный обзор всех соединений, разбитый по частям для простоты понимания.

Для иллюстрации подобных вещей лучшим средством является *Patchage*; запустите его и всё проверьте — вам поможет цветовой код: аудиопорты синие, а MIDI-порты — красные. Все менеджеры подключений имеют свои достоинства и недостатки. Хоть *Patchage* и очень прост при работе с малым числом входов и выходов, он становится крайне запутанным при изобилии соединений. В этом случае хорош встроенный в *Ardour* матричный менеджер подключений, поскольку вы имеете с ними дело через систематизированные вкладки. Так будет аккуратнее.

Это простая сессия, но она демонстрирует, как аудио- и MIDI-соединения работают в JACK. Она также включает внешние приложения DAW, одно из которых синхронизировано с JACK. Попробуйте все менеджеры JACK/подключений и решите, который удобнее для вашего рабочего процесса. Они все выполняют одну задачу, просто немного разными путями. Но для создания одних и тех же соединений логика одна и та же.

Лучшее в JACK то, что он не запирает вас в рамках одного рабочего процесса. Мы увидели, как синхронизировать программы и создавать соединения между ними, но если вы хотите делать всю работу в одной DAW, не покидая её — возможно и это. JACK вас ничем не сковывает, оставляя открытыми все опции, чтобы в любой момент извлекать из них максимум преимуществ, и допускает различные варианты рабочего процесса, будь то монолитный или модульный. Понравившийся вам инструмент JACK настроит согласно любой вашей фантазии. LXF

Модульность

Модульные структуры — это когда вы используете более одной программы. JACK позволяет соединять и синхронизировать самые разные программы аудио, и вы можете извлечь лучшее из каждого приложения. Для управления и переназначения в сложных структурах можно использовать управление сессией. Одним из комплектов модульных приложений является *Non Suite*, включающий *Non Timeline*, *Non Mixer* и *Non Sequencer*, хотя, по сути, в модульную структуру можно ввести любое JACK-совместимое приложение. *Non Suite* также содержит очень понятный менеджер сессии, под названием *Non Session Manager*.

Строим ВМ для разработки PHP

Кент Ельчук рассказывает о виртуальных машинах (ВМ) и скриптах PHP для создания идеальной среды разработки.

Может быть, PHP и не отличается красотой, но уж точно полезен, и все, что пригодно для разработки и изучения PHP — хорошо. Мы выясним, как применение полной ВМ [VM] с нуля поможет в создании web-страниц, запуске скриптов PHP и доставке электронной почты. И кроме того, мы покажем вам, как клонировать виртуальную машину и перенести ее на другую машину, независимо от того, будет ли хост или гость 32- или 64-битным. Мы обсудим весь процесс сборки — от скачивания самой свежей версии Ubuntu, установки *Virtualbox* на Ubuntu, установки пакетов и модификации пакетов.

Хотя данный урок сосредоточен на *Virtualbox*, в области виртуализации имеются альтернативы, например, *KVM*, *VMware* и *Xen*. Виртуальные машины бывают весьма удобны и разноплановы. Например, можно использовать пакеты, подобные *Rsync*, и другие методы для синхронизации файлов и баз данных с другими виртуальными машинами или непосредственно с другой машиной.

Установка Virtualbox

Чтобы установить *Virtualbox* на машинах на базе Debian, надо добавить строку в файл **/etc/apt/sources.list** для соответствующего пакета. Хотя *Virtualbox* отличается гибкостью, вам, возможно, придется включить виртуализацию в BIOS для нормальной работы с 64-битными гостями.

Пример установки показан ниже. Для начинающих, откройте файл **sources.list** в редакторе. Все команды запускаются пользователем root:

```
vi /etc/apt/sources.list
```

Скопируйте в файл **sources.list** строку

```
deb http://download.virtualbox.org/virtualbox/debian quantal contrib.
```

Затем запустите следующие команды:

```
wget -q https://www.virtualbox.org/download/oracle_vbox.asc -O- | sudo apt-key add -
apt-get update
apt-get install virtualbox-4.3
```

На запросы отвечайте Y. В случае неполадок загляните в официальную wiki на http://bit.ly/VirtualBoxDLs.

В более старых установках **sources.list** может стать проблемой. В качестве альтернативы вы всегда можете сделать резервную копию и просто



› Создавая новую ВМ, задайте ей имя.

добавить исходник для *Virtualbox*, или обновить файл **sources.list**. И всегда можно сгенерировать список: http://repogen.simplylinux.ch.

Чтобы открыть *Virtualbox*, найдите его в Dash или наберите **virtualbox** в командной строке, а чтобы создать новую машину, выберите New, укажите Name, выберите Type, Version и нажмите Next. После этого нужно указать размер памяти или просто принять выбор по умолчанию. Эти параметры можно изменить после установки. Выберите Next > Create.

Когда вам предложат выбрать тип файла, хорошей опцией будет VDI, это также настройка по умолчанию. Закончите установку и создайте виртуальную машину. По умолчанию размер диска часто составляет 8 ГБ, а на создание виртуального сервера Ubuntu может потребоваться чуть больше места на диске.

После установки выберите свою виртуальную машину из списка. После этого нажмите на Settings > Storage > Empty (Under Controller IDE) > Select your ISO > OK. Теперь вы можете запустить свою машину, просто выбрав Start, что очень удобно. Как и в обычном ПК, вы проходите процедуру установки, и в случае Ubuntu нужно выбрать Install Ubuntu. Хотя вся установка ничем не отличается внешне от установки на жесткий диск, это всего лишь виртуальная машина, располагающаяся в файле **.vdi**. Установив Ubuntu, вы сможете запускать ВМ и входить через созданные вами имя пользователя и пароль.

Настройка View дает вам разные опции. В новой инсталляции *Virtualbox* внешний вид экрана вам может не понравиться. Хорошая новость в том, что это довольно легко изменить. Чтобы у вас было еще больше опций View, таких, как Seamless Mode, установите на гостевой операционной системе гостевые дополнения *Virtualbox*. Их можно скачать с http://download.virtualbox.org/virtualbox. Учтите: все это нужно запускать от имени администратора.

Теперь настройте желаемый вид и запаситесь терпением. Сначала экран может показаться маленьким, но это исправимо. После установки дополнения Vbox Guest вы получите доступ к Seamless Mode, Full Screen и Scaled Mode, однако вам может понадобиться одна-две перезагрузки, чтобы получить все то, что вы хотели.

Настройка клонов и сервера

Чтобы клонировать виртуальную машину, выберите machine > Snapshots > Click the Clone icon (которая выглядит как страницы или как овца). В этот момент нужно дать клону имя и выполнить ряд очевидных шагов. У вас есть опции перемещения ВМ на новый ПК. Одним из способов будет переместить всю папку виртуальной машины из папки *Virtualbox*, VMS. Потом просто запустите *Virtualbox* и выберите Machine > Add. Другой способ — создать новую машину и добавить существующий файл VDI. Теперь, когда *Virtualbox* настроен, давайте запустим функциональный сервер с *Apache*, *MySQL*, PHP и Postfix. Эта установка займет всего несколько минут.

Для установки *Apache* скомандуйте

```
apt-get install apache2
```

от имени root. Установив *Apache*, откройте браузер и введите localhost. Перед вами должна открыться страница по умолчанию Apache2 Ubuntu. Для *MySQL* запустите команду

```
apt-get install mysql-server mysql-client
```

от имени root. Во время установки вам несколько раз предложат ввести пароль. Можете так и сделать — или просто нажать Enter, продолжив установку без создания пароля. Если вы решите обойтись без пароля, а потом передумаете, пароль *MySQL* меняется легко, всего одной командой. Далее, установим PHP, опять же как root, командой

```
apt-get install php5 libapache2-mod-php5
```

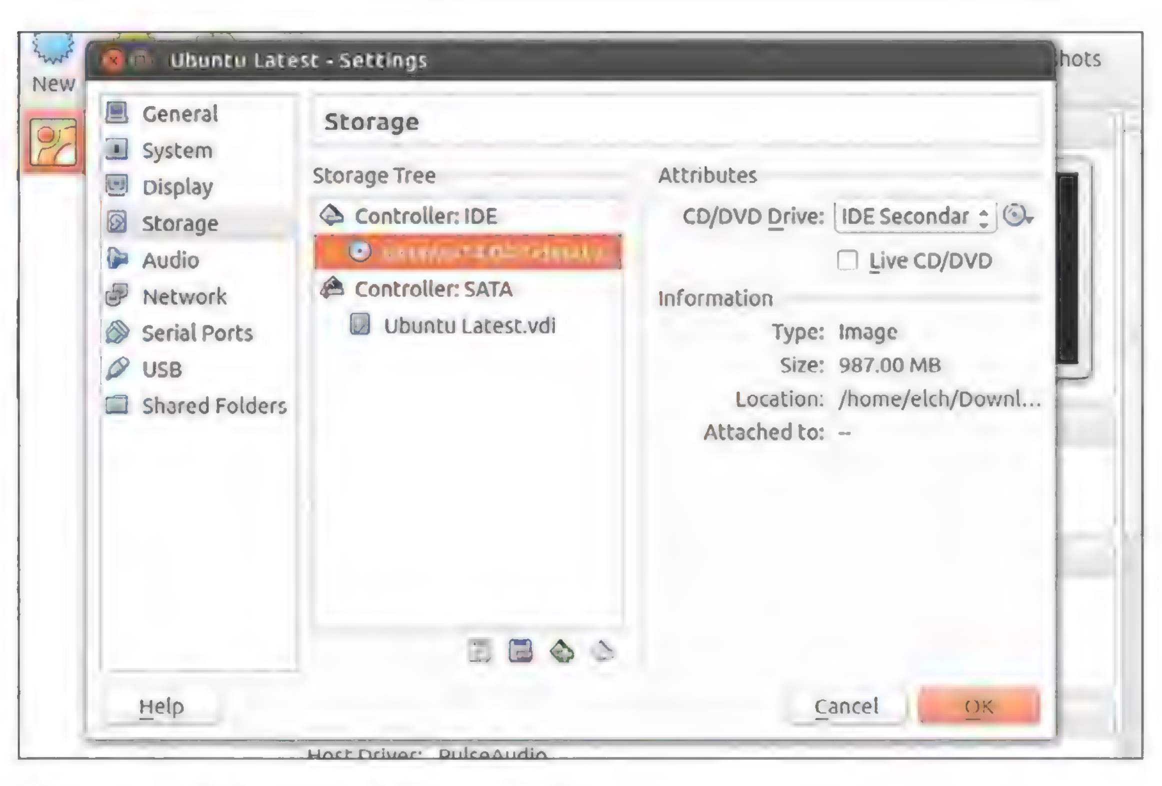
На время написания, папкой, отображающей сайт, была **/var/www/html**. В предыдущих

инсталляциях папкой по умолчанию была **/var/www**. Учтите также, что *Apache* устанавливает AllowOverride на None, и, возможно, вы решите сменить это на All, если хотите использовать файлы **.htaccess**.

Давайте пойдем дальше и создадим тестовый файл PHP, чтобы посмотреть, как он работает. Вы можете открыть любой редактор. По умолчанию редакторы *Nano* и *Gedit* готовы к работе. Желая использовать *vi*, запустите команду

```
apt-get install vim
```

Вам нужно будет создать файл под названием **test.php**. Конечно же, он помещается в папку **/var/www/html**, о которой мы упоминали выше. Простая строка кода — **<?php echo phpinfo();?>**. Эта функция покажет вам множество подробностей настройки. Теперь вы можете открыть страницу в своем браузере и узнать подробности своей настройки PHP. Эту настройку можно изменить, отредактировав настройки по умолчанию в **/etc/php5/apache2/php.ini**. Вы можете легко войти как root и отредактировать этот файл.



› **Выберите для своей виртуальной машины ISO-образ.**

DNS и переадресация портов

PHP требует установки, но другие языки скриптов, типа Python и Perl, уже есть по умолчанию. Об этом вам сообщат **whereis python** и **whereis perl**.

На данный момент сервер работает, и нужно только немного его поднастроить, чтобы его было видно по такому адресу, как example.com. Если вы хотите использовать доменное имя, вам придется направить DNS на ваш IP. Для этого вы можете использовать бесплатный DNS-сервис или свой сервис web-хостинга. Технически, вы создаете запись А, которая указывает на IP-адрес, предоставленный вам вашим интернет-провайдером (ISP).

После настройки DNS нужно настроить ваш роутер, а чтобы отображался ваш сайт, включить переадресацию портов на порт 80.

По умолчанию *Apache* и другие серверы используют после установки именно этот порт. Однако помните, что переадресации портов вы можете добавить. Например, чтобы электронная почта переадресовывалась на порт 25, SSH — на порт 22, и, опционально, FTP на порт 21. Все эти правила переадресации будут использовать один и тот же локальный сетевой IP-адрес вашей ВМ.

Хорошо, а *какой* у вас локальный IP-адрес? Когда вы создаете виртуальную машину с помощью *Virtualbox*, ваше сетевое окружение будет использовать NAT и по умолчанию примет IP-адрес вашего соединения с хостами. Однако вы можете настроить *Virtualbox* так, чтобы у виртуальной машины был собственный IP-адрес. Вы можете изменить его в любое время перед загрузкой своей виртуальной машины. Чтобы изменить настройки, щелкните правой кнопкой по своей ВМ и щелкните по Settings > Network > Attached to.

Соединение через Ethernet и NAT — надежное и быстрое; или используйте соединения Bridged с Ethernet (eth0), изменив неизбирательный режим [promiscuous mode] на Allow VMs. Последний способ позволяет вашей ВМ использовать Ethernet и получить собственный IP. Если вы запустите **ifconfig** на хосте и госте, вы должны увидеть два разных IP-адреса. Роутер тоже должен их увидеть. Хотя вы можете добавить беспроводные адаптеры USB и использовать *Virtualbox* беспроводным методом, наше руководство будет рассматривать Ethernet, в котором нет обычных для беспроводного соединения проблем.

Соорудив рабочий сервер, способный показывать публичные web-страницы, вы, возможно, решите пойти дальше: добавить сюда функции электронной почты. Как и на арендованном VPS [виртуальный частный сервер] или выделенной машине от хостинг-провайдера, у вас есть базовые инструменты, приличествующие любому владельцу сайта.

Теперь мы, наконец, можем перейти к учебнику по PHP.

Вот он, учебник по PHP

Итак, у вас все рассажены по лавочкам; можете открыть свой любимый редактор и начать кодировать на PHP. Хотя у PHP есть противники, его популярность является своего рода феноменом языка скриптов серверной стороны, и большинство его пользователей работают с PHP на каком-то сервере Linux. Среди web-дизайнеров и web-разработчиков также очень распространено использование платформы хостинга с набором LAMP (Linux, *Apache*, *MySQL* и PHP) или LEMP. (*Nginx*, HTTP-сервер и обратный прокси-сервер, заменяет *Apache*. *Nginx* произносится “Engine-X”, отсюда и буква E вместо N.)

Это привело к созданию изрядного числа скриптов и библиотек, с легкостью удовлетворяющих множеству потребностей. Среди хорошо известных приложений — *WordPress*, *Joomla*, *Drupal*, *Magento*, *Prestashop* и масса других скриптов. »

Безопасность PHP

В вашей новой установке PHP нет безопасного режима [safe mode], и есть проблемы с безопасностью. Но для домашнего сервера вы можете устранить большую часть проблем, отключив **file_uploads**, отключив функции, контролируя весь доступ к системе и проводя мониторинг всех изменений файлов с помощью скриптов *Bash*.

Все изменения в вашей настройке PHP можно сделать в файле **/etc/php5/apache2/php.ini**. Например, вы можете изменить настройку **file_uploads** с On на Off и отключить побольше функций.

По умолчанию, функции вроде exec, shell_exec и base64 при наличии файлов в вашей системе могут создать немалую головную боль.

Такие функции, как shell_exec и exec, позволяют записывать команды Linux, способные перезаписать файлы, показать публике необработанный код командой **cat** и даже взять на себя контроль за базами данных. Сеть полна коварных скриптов, которые могут демонстрировать и менять ваши данные.

При использовании PHP возникают проблемы не только с безопасностью. Плохой код может подвергнуться эксплойту, и в ваши web-директории попадут вредоносные файлы **PHP.ini**, имеющие приоритет перед основным файлом. От этого можно оборониться заплаткой suhosin.

В общем и целом, с вашим домашним PHP сервером все будет хорошо, если вы отключите выгрузки и запретите доступ к своей системе. И, конечно же, никогда не доверяйте тому, что пользователи вписывают в формы. Есть множество функций типа **htmlentities()** и **mysql_real_escape_string()**, «дезинфицирующих» пользовательский ввод.

Помимо скриптов, есть также масса сред, позволяющих использовать библиотеки для написания кода, например, Symphony, Codeigniter и Zend.

Хотя PHP удобен, он настолько расхлябан, что написать на нем плохой код очень легко. С другой стороны, вы можете создать солидные объекты и солидный код. Кроме того, есть множество библиотек, которые можно добавить, чтобы отделить логику от дизайна; как движок шаблонов Smarty. Если у вас есть опыт работы с C или Java, у вас есть должная основа для написания солидного кода, минус необходимость декларировать типы переменных.

Популярных скриптов полным-полно, но у многих из них имеется темная сторона. Некоторые очень велики, и стали настолько популярны, что превратились также и в популярные мишени для взломщиков. С другой стороны, если вы пишете простые web-сайты PHP с упором на безопасность, код становится намного лучше управляемым, а производительность — превосходной.

На это может уйти некоторое время, но при наличии желания и терпения вы сможете достичь такого уровня кодирования в PHP, *MySQL*, HTML и CSS, который позволит вам создавать все, чего душа пожелает (со временем, конечно).

Создание скриптов PHP

В этом разделе мы рассмотрим создание скриптов PHP, что делается через командную строку или с помощью файлов в браузере. PHP интерпретируется на сервере и отображает результат в браузере пользователя. В основном это именно то, что вы видите ежедневно на сайтах, использующих приложения PHP, типа *WordPress* или *Magento*.

Если вы создаете скрипты в безопасной директории на вашей машине Linux вне директории **home**, вы можете смело запускать эти файлы через командную строку. Иногда вам нужны только скрипты-утилиты, запускаемые как задание *Cron* или когда вы захотите сами их выполнить.

Давайте рассмотрим некоторые базовые моменты — например, как можно создавать комментарии, переменные, массивы, циклы, функции,



› Установите свою виртуальную машину так же, как если бы вы делали установку на жесткий диск.

классы и объекты, CRUD, запросы *MySQL* и включать файлы.

Все файлы PHP заканчиваются расширением **.php**. Любой код между тэгами **<?php?>** будет интерпретироваться. Любой код за пределами этих тэгов будет интерпретирован как HTML в браузере. Тэги **<??>** тоже работают, но только если вы настроили PHP (**php.ini**) на разрешение использования коротких тэгов.

В PHP есть три метода комментирования (то есть добавления примечаний, которые не интерпретируются, чтобы вы легко могли следить за тем, что делает код):

```
// Комментарии после двух прямых слэшей
# Команда после символа решетки
/* Комментарии внутри таких ограничителей
могут иметь произвольное количество строк */
```

Общая переменная — это либо строка (символы), либо число (цифры). Переменные можно декларировать через символ $. В приведенном ниже примере видно, как декларировать оба типа переменных:

```
$my_string = "My string";
$my_number = 3;
```

Вы можете выводить печать на страницу, заключив текст в одиночные или двойные кавычки. Одиночные означают «как есть», а двойные будут интерпретироваться. Это означает, что вы можете напечатать переменные в одиночных кавычках:

```
echo 'Привет!'; // выводит: Привет!
echo ' Привет $variable'; // выводит: Привет $variable
```

...и в двойных кавычках:

```
$variable = "Джон";
echo "Привет!"; // выводит: Привет!
echo "Привет, $variable"; // выводит: Привет, Джон
```

Бывают случаи, когда вы видите двойные кавычки внутри двойных кавычек — например, при использовании HTML. Если вы столкнулись с таким случаем, лучше сделать обратный слэш, чтобы избежать интерпретации. Вот пример:

```
echo "<a href =\"example.com\">My Link</a>"
```

В качестве альтернативы всему вышеперечисленному можно заменить двойные кавычки одиночными или использовать иной вид записи:

```
<?php ?>
<a href ="example.com">My Link</a>
<?php //Add php here ?>
```

Примеры функций

Те, у кого есть опыт работы с C или Python, знают, что такое печать и функция **sprintf()**. Как ею пользоваться, показано в примере ниже.

```
$variable1 = 10;
$variable2 = 5;
$variable3 = $variable1 + $variable2;
$variable4 = 'в футбольной команде';
$format = 'Состав %d мужчин и %d женщин,
всего %d человек %s.';
echo sprintf($format, $variable1, $variable2,
$variable3, $variable4);
```

Массивы — это группы элементов. Синтаксис индексированного и ассоциативного массивов показаны ниже:

```
$indexed = array('apple', 'peach','pear', 'plum');
$associative = array('name' => 'John', 'age' => 24,
'height' = > '2 metres');
```

Две команды, которые всегда пригодятся — **print_r()** и **var_dump()**. Они покажут вам ключи и значения массивов, если вам понадобятся подробности. Попробуйте набрать **print_r($indexed)** и **print_r($associative)**, чтобы сравнить ключи и значения обоих массивов.

В PHP также есть множество функций над массивами, выполняющих их сортировку, добавление целочисленных значений и многое другое. Переменные и блоки текста разделены точкой, и это называется конкатенацией. Вы увидите конкатенацию в действии в параграфе, посвященном циклам (ниже).

Среди самых общих циклов, которые вам встретятся и которыми вы будете пользоваться — циклы foreach, for и while. Обычно цикл foreach анализирует массив. Вы можете разделить любой массив на значения или на ключи и значения. Примеры разных вариантов циклов и функций показаны ниже.

» Цикл Foreach Этот цикл анализирует массив по порядку.

```
foreach($indexed as $item){echo $item . "<br/>";
// конкатенация переменной и тэга HTML <br/>.
}
foreach($associative as $key => $item){
echo $key. " - " . $item . "<br/>";
```

Скрипты PHP и командная строка

Запуск скриптов PHP из командной строки несколько отличается от их интерпретации в браузере. Главное отличие состоит в том, что '\n' оставляет место для новой строки в командной строке, а '
' делает то же самое, когда HTML интерпретируется в браузере. Кроме того, когда PHP работает со скриптами в командной строке, вы добавляете следующий код в начале файла:

```
#!/usr/bin/php
```

а для исполнения файла в текущей директории командуете

```
php ./update-status2.php
```

Вот типичное задание *Cron*, которое запускает файл **filename.php** каждые десять минут. Доступ к файлу *Cron* для root или других пользователей можно получить на **/var/spool/cron/username** или просто через команду **crontab -e**.

```
*/10 * * * * username /usr/bin/php -f /var/
www/html/filename.php
```

Файлы, запускаемые через задания *Cron*, могут требовать другого кодирования по сравнению с файлами, запускаемыми из браузера. Файлы в браузере относятся к папке **public_html** или **www**. А файлы, запускаемые через *Cron*, требуют, чтобы все было включено и чтобы файлы выбирали путь, относящийся к папке root.

Поэтому, хотя **include ("my_included_file.php")** будет работать в браузере, если файл находится в той же папке, файлу *Cron* нужен путь по типу **include ("/var/www/html/my_included_file.php")**, чтобы суметь определить местоположение.

Альтернативой смене имени пути будет удаление функций include или require и размещение всего кода в одном файле.

```
// конкатенация переменных и печатаемого текста.
}
```

» Цикл For Следующий цикл For печатает номер один и добавляет 1 каждый раз, когда повторяется. Он продолжается, пока значение $i меньше 5.

```
for($i=1; $i < 5; $i++) {
echo $i; // выводит: 1234
}
```

» Цикл While Следующий цикл печатает номер один и добавляет 1 каждый раз, когда повторяется. Он тоже продолжается, пока значение $i не доходит до 5.

```
$i = 1;
while($i < 5) {
echo $i; // выводит: 1234
$i++;
}
```

» Include Files Если вам надо добавить еще один файл к текущему файлу, это можно сделать с помощью функций **include()**, **include_once()**, **require()** и **require_once()**. Пример показан ниже.

```
include("myfile.inc");
```

» Функции Это наборы процедур, которые можно вызвать. Функции вызываются по имени. Обычно функции располагаются в собственных файлах и включаются в файл, где вы хотите их использовать. Простая индивидуально настраиваемая функция показана ниже, и ее можно исполнить с помощью **name_it()**:

```
function name_it (){
echo "Here is my text";
}
name_it();
```

Поверх всего этого PHP устанавливается с тысячами функций. Некоторые из них — объектно-ориентированные, например, **new DOMDocument()**. http://PHP.net — сайт, на который стоит заходить по всем вопросам, связанным с PHP. Например, на странице http://php.net/manual/en/filter.filters.php показано множество фильтров проверки достоверности, которые можно использовать с функцией, например, **filter_var()**, применяемой в таких средах, как Codeigniter.

Используя функции PHP, помните, что они-таки потребляют память, и некоторые — довольно много. Кодируя с целью увеличить эффективность, можно минимизировать потребление ресурсов и добиться более высокой производительности.

» Классы и Объекты В PHP вы можете создавать код процедурно сверху вниз. Но ООП (Объектно-ориентированное программирование) дает куда более управляемый способ написания кода. Основной класс является родительским, и у него могут быть дочерние классы, которые могут полностью или частично наследовать родительскому классу. Чтобы использовать ООП, вы создаете класс, который включает свойства и методы. Свойство подобно обычной переменной, а метод подобен функции. Чтобы приписать значение

PHP Version 5.5.9-1ubuntu4.3

System	Linux ubuntu-VirtualBox 3.13.0-32-generic #57-Ubuntu SMP Tue Jul 15 03:51:12 UTC 2014 i686
Build Date	Jul 7 2014 16:30:18
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php5/apache2
Loaded Configuration File	/etc/php5/apache2/php.ini
Scan this dir for additional .ini files	/etc/php5/apache2/conf.d
Additional .ini files parsed	/etc/php5/apache2/conf.d/05-opcache.ini, /etc/php5/apache2/conf.d/10-pdo.ini, /etc/php5/apache2/conf.d/20-json.ini, /etc/php5/apache2/conf.d/20-readline.ini
PHP API	20121113
PHP Extension	20121212
Zend Extension	220121212
Zend Extension Build	API220121212,NTS
PHP Extension Build	API20121212,NTS
Debug Build	no
Thread Safety	disabled
Zend Signal Handling	disabled
Zend Memory Manager	enabled
Zend Multibyte Support	provided by mbstring
IPv6 Support	enabled
DTrace Support	enabled
Registered PHP Streams	https, ftps, compress.zlib, compress.bzip2, php, file, glob, data, http, ftp, phar, zip
Registered Stream Socket Transports	tcp, udp, unix, udg, ssl, sslv3, tls
Registered Stream Filters	zlib.*, bzip2.*, convert.iconv.*, string.rot13, string.toupper, string.tolower, string.strip_tags, convert.*, consumed, dechunk

This program makes use of the Zend Scripting Language Engine.
Zend Engine v2.5.0, Copyright (c) 1998-2014 Zend Technologies
with Zend OPcache v7.0.3, Copyright (c) 1999-2014, by Zend Technologies

Powered by Zend Engine 2

› php.info.php показывает ваш PHP config, и его можно изменить, отредактировав файл php.ini или обойти, добавив файл php.ini в папку html.

классу, вы используете «новое» ключевое слово. Свойства и методы бывают публичными, приватными и защищенными. «Публичные» означает, что доступ к ним можно получать отовсюду; «приватные» доступны только для определенного класса, а «защищенные» доступны для родительского и дочернего класса, где они были объявлены. Еще один аспект ООП — использование ключевого слова **$this** для ссылки на свойство. Кроме того, после приписывания значения объекту вы используете **->** для доступа к методам. Пример ниже должен это прояснить.

```
Class Test {
public $name = 'John';
function get_name(){
return $this->name;
}
}
$my_test = new Test();
echo $my_test->get_name();
```

» Запросы *MySQL* Очень полезная функция — запросы *MySQL*, которые возвращают нужные вам данные. Запросы можно исполнять с помощью PDO или функции **mysql_query()**. Чтобы сделать запрос, вы соединяетесь с базой данных посредством аутентификации.

» CRUD Это сокращение от Create, Read, Update and Delete. Большинство web-приложений используют CRUD, написанный так или иначе, чтобы позволить пользователю вносить изменения в базу данных. Все события CRUD происходят с использованием команд *MySQL* **SELECT**, **INSERT**, **UPDATE** и **DELETE**.

» Запуск PHP из *Bash* Велики шансы того, что у вас есть опыт работы со скриптами *Bash*, и вы можете использовать этот опыт, добавив код PHP в скрипты *Bash* с помощью тэгов Here Document('EOF').

```
#!/bin/sh
php_cwd=`/usr/bin/php << 'EOF'
<?php echo getcwd(); ?>
EOF
echo "$php_cwd"
```

Теперь у вас есть основы для написания программ PHP на вашей новой ВМ. Можете сделать ее копию и развернуть ее где пожелаете. Кроме резервного копирования *Virtualbox*, можно применять **ssh**, **rsync**, **mysqldump**, **scp** и **cron**, чтобы у вас на другой машине всегда была под рукой резервная копия. Большинство сайтов и web-приложений PHP используют файлы и базы данных. Если вы хотите клонировать или синхронизировать эти источники данных вашей ВМ, оно будет очень удобно и быстро реализуемо. Наслаждайтесь. LXF

Сервер электронной почты

Чтобы приступить к работе с электронной почтой, надо установить почтовый сервер. Их несколько; примеры — *Postfix* и *Exim*. В этом руководстве мы воспользуемся *Postfix*.

Для установки *Postfix* запустите команду, как показано ниже.

```
root# apt-get install postfix postfix-mysql
dovecot-core dovecot-imapd dovecot-lmtpd
dovecot-mysql dovecot-pop3d
```

Во время установки вам, возможно, придется нажать на клавишу tab > Internet Site > нажать на клавишу tab и выбрать OK > Нажать Enter > Give system mail name or use the default [дать имя системной почте или использовать предложенное по умолчанию] > нажать tab > Enter.

После успешной настройки сервера надо будет отредактировать файл **/etc/postfix/main.cf** и добавить скрипт, например, *Squirrelmail*, для обработки входящих и исходящих сообщений. Помните: настройка электронной почты может потребовать немалого времени, и эта задача выходит за рамки нашего руководства.

Однако вы можете проверить исходящую почту простыми командами, приведенными ниже. Чтобы проверить исходящую почту, выпишите каждую строку и нажмите Enter.

Когда вы дойдете до точки после тела почты (текста сообщения), введите ее и нажмите Enter так же, как вы проделывали для других команд. Это не опечатка, таково требование для отправки почты.

```
telnet localhost 25
HELO mail.example.com
MAIL FROM: test@example.com
RCPT TO: chukweb@shaw.ca
DATA
Subject: Добавьте тему
Введите здесь текст сообщения
.
QUIT
```



По рецептам доктора Брауна

Д-р Крис Браун
Доктор обучает, пишет и консультирует по Linux. Ученая степень по физике элементарных частиц ему в этом совсем не помогает.

Эзотерическое системное администрирование из причудливых заворотов кишок серверной
















Яйца и корзины

Президент был, мягко говоря, обеспокоен. Во-первых, он только что сообразил: доклад, который он согласился прочесть на Международном симпозиуме по цифровому биоразнообразию, помешает ему в это самое время посетить бейсбольный матч сына. Втайне он считал упомянутый симпозиум сборищем психов, но советники убедили его, что зато он получит их голоса. А во-вторых, его озадачили 46 только что полученных сообщений о двухуровневой авторизации, и все они поступили, похоже, от него самого.

Президент решил расставить должные приоритеты и снял трубку телефона. «Элис, напиши Бобу, пусть он отменит этот доклад по цифровому биоразнообразию». «Извините, босс, но у меня возникли проблемы с компьютером. Только что звонили Боб и Ева, у них тоже какие-то проблемы». Короткая прогулка по коридору показала, что в веренице офисов пользователи проклинают свои компьютеры громче, чем обычно.

Президент позвонил председателю Совета национальной безопасности. «Позовите к телефону Наделлу [исполнитель обязанностей генерального директора Microsoft с февраля 2014 г., — *прим. пер.*]», — потребовал он. Нет смысла быть могущественнейшим человеком на планете, полагал президент, если ты не в состоянии добраться до самого верха. Но в Microsoft никто не отозвался. По неизвестной причине Microsoft, кажется, полностью вымерла.

Чем больше людей звонили и говорили, что у них проблемы с компьютером, тем менее могущественным чувствовал себя президент. Наконец, в офис вбежал его младший помощник, запыхавшийся после бега по четырем лестничным пролетам. «Господин президент, сэр. Думаю, вы должны знать, что в подвале есть один тип, у которого компьютер все еще работает. У него установлен Linux».

chris.linuxformat@gmail.com

Взлом паролей

Что общего у беличьего шума и второго закона термодинамики? Ответ кроется в энтропии.

Энтропия — мера случайности или беспорядка. Второй закон термодинамики утверждает, что энтропия замкнутой системы никогда не уменьшается. (Представьте себе, например, комнату подростка). Моя любимая серия карикатур «Дилберт» — та, в которой босс говорит: «Начиная с сегодняшнего дня, все пароли должны содержать буквы, цифры, каракули, язык жестов и беличий шум». Как и во всех сериях «Дилберта», в любом безумии есть доля правды.

Дело в том, что чем шире набор символов, из которых составляются пароли, тем выше их энтропия и тем больше времени потребуется на их взлом методом перебора. Я немного поиграл с калькулятором перебора от Open Security Research (http://bit.ly/BruteForceCalc) и привел результаты в таблице (ниже). Расчеты основаны на чистой комбинаторике и предполагают, что вы можете проверять по 1 000 000 паролей в секунду. Реальная цифра будет зависеть от используемого алгоритма и от того, сколько воды вы можете вылить, охлаждая свой суперкомпьютер. Но итог один: более длинные пароли и более широкий набор символов существенно меняют дело.

Но Марк Бернетт [Mark Burnett], который потратил, кажется, полжизни на сбор урожая в виде логинов и паролей (как он уверяет, «из источников, которые уже были сделаны публичными»), опубликовал на Xato.net (http://bit.ly/10KTopPasswords) такие данные, которые убедят вас в том, что вся эта история с беличьим шумом и комбинаторикой не играет никакой роли.

По данным Марка, 40% всех паролей попадают в список Top 100, 91% — в Top 1000 и впечатляющие 98,8% — в Top 10000. Конечно, результаты зависят от того, где собираются данные. Марк признает, что эти пароли были получены в основном с сайтов, которые не требуют от пользователей надежных паролей. Так что забудьте о беличьем шуме. А на XKCD (https://xkcd.com/936) есть прекрасный пример, в котором энтропия типичного пароля, полученного путем ритуального потрошения настоящего слова (Tr0ub4dor&3), сравнивается с энтропией пароля, полученного из четырех несвязанных слов (correct horse battery staple). В последнем на 16 бит энтропии больше, а значит, взломать его в 65000 раз сложнее, но гораздо проще запомнить и набрать.

Длина	Набор символов	Время на взлом методом перебора
8	Цифры от 0 до 9	111 секунд
10		185 минут
12		308 часов
8	Буквы верхнего регистра A–Z	60 часов
10		5 лет
12		3000 лет
8	A–Z a–z 0–9 !@#$%^&()-_+=	60 часов
10		206000 лет
12		1 млрд лет

WINE

Запускайте программы Windows на Linux в *WINE*. Добрый Доктор помогает вытащить пробку из бутылки и любезно отпивает глоток-другой...

Вы системный администратор, и ваша компания только что сообщила о переводе всех своих операций с Windows на Linux. Так как вы продвигали эту идею уже почти десять лет, это выглядит как наконец-то сбывшаяся мечта. Но начальная эйфория ослабевает, когда к вам зачастят пользователи с опрокинутыми лицами, говоря, что им не обойтись без приложения А или В, а оно запускается только в Windows.

К таким заявлениям надо относиться скептически. Лишь для очень немногих задач нет открытого аналога, который не уступал бы своим проприетарным братьям. Возможно, эти пользователи просто не хотят выходить из своей зоны комфорта или изучать новые технологии. Или их беспокоит, что придется обмениваться документами с другими компаниями, еще приверженными проприетарным форматам. А возможно, юная Рози из бухгалтерии серьезно подсела на *World of Warcraft* и угрожает перестать платить вам зарплату, если больше не сможет играть на работе.

Если вам нужно запускать приложения и для Windows, и для Linux, есть три варианта: настроить двойную загрузку, запускать Windows в виртуальной машине из Linux (или наоборот — неважно) или воспользоваться *WINE*.

По сути, *WINE* — это слой совместимости, поддерживающий API, которое имитирует стандартные системные DLL Windows: USER32, GDI32 и KERNEL32. Этот слой совместимости располагается поверх POSIX-совместимого ядра (которое работает в Linux, BSD, Solaris и Mac OS X). С точки зрения приложений Windows, *WINE* — это эмуляция системы Windows, и некоторые говорят, что *WINE* расшифровывается как “Windows Emulator”. Парням из штаба *WINE* это не нравится: они предпочитают, чтобы *WINE* было сокращением от “*WINE* Is Not an Emulator [*WINE* — это не эмулятор]”. Их беспокоит то, что эмуляция для многих означает плохую производительность.

Не будем обольщать вас: с *WINE* ладят не все приложения Windows. Эмуляция несовершенна, и одни приложения запускаются лучше, другие хуже, а некоторые не запускаются вообще. На сайте www.winehq.org есть огромная база данных, где каждое приложение оценивается по шкале Platinum, Gold, Sliver, Bronze или Garbage. Игры здесь лидируют. В лучших десятках платиновых, золотых и серебряных приложений (итого 30 приложений) игры занимают 24 места. Оценка часто зависит от используемой версии программы и библиотеки *WINE*. Например, оценки *Microsoft Money* разнятся от Platinum до Garbage [Хлам] в зависимости от версии. На момент написания статьи в списке 21 626 приложений, но многие из них старые, и многие ссылки на сайты поставщиков или разработчиков не работают.

Первый глоток *WINE*

Так или иначе, откупорим бутылку и сделаем глоток (или два) *WINE*. Я предпочел делать это в Ubuntu 14.04, и оказалось, что *WINE* есть в репозиториях Ubuntu, поэтому я смог установить ее просто командой

```
$ sudo apt-get install wine
```

Придется-таки подождать, потому что установка включает в общей сложности 174 пакета. В ходе установки вас попросят принять лицензионное соглашение для шрифтов TrueType от Microsoft. Установка на RedHat-производный дистрибутив, вроде CentOS, потребует чуть больше работы, так как нужно включить репозитории EPEL (инструкции см. во врезке «EPEL» на стр. 58).

Установив *WINE*, вы сможете запускать исполняемые файлы Windows с помощью команды **wine**. В *Wine* включены несколько программ, включая *Notepad*, *Wordpad*, *Regedit*, интерпретатор команд (*cmd*) и *WINE*-версия *Internet Explorer*. Можете попробовать запустить их сразу командой вроде

```
$ wine wordpad
```

Вообще-то у некоторых из этих программ есть скрипты-обертки в **/usr/bin**, поэтому их можно запускать по имени, как любую другую команду:

»

WINE или виртуализация?

Виртуализация (запуск виртуальной машины с Windows на компьютере с Linux) позволит запускать приложения для Windows в Linux гораздо проще. Так вы получите «настоящее» Windows-окружение, и большинство приложений должны запуститься. В случае с *WINE* поддержка Windows-приложений неравномерна. С другой стороны, у *WINE* гораздо меньше требования к памяти, так как в этом случае у нас нет полностью установленной системы Windows.

Но важнее то, что в случае с *WINE* вам не требуется лицензионная копия Windows. Документ Microsoft с описанием прав на запуск Windows в виртуальных машинах занимает шесть страниц, и его совершенно невозможно понять, если у вас нет ученой степени по бесконечно дифференцируемым лицензионным соглашениям, но ясно одно: это не бесплатно. Помните, что даже с *WINE* у вас должна быть легальная копия программы, которую вы запускаете.

› Неопровержимое доказательство того, что 2+2 = 4. *Microsoft Excel* и *LibreOffice Calc* бок о бок работают в CentOS.



› **Главное окно *Crossover*. В нем можно запускать приложения Windows и устанавливать новые, а также управлять бутылями.**

```
$ notepad
```

Текстовые консольные приложения запускаются командой **wineconsole**, причем они открываются в отдельном консольном окне. Так, при запуске

```
$ wineconsole cmd
```

у вас появится ваш личный интерпретатор команд Windows. Повезло! *WINE* создает в каталоге **~/.wine** (по умолчанию) небольшое Windows-подобное окружение. Например, в каталоге **~/.wine/drive_c/users/chris** у меня есть миниатюрный каталог home в стиле Windows, где символические ссылки с традиционными именами Windows (например, "My Pictures"), указывающие на их Linux-эквиваленты (например, **/home/chris/Pictures**). В каталоге **~/.wine/drive_c/windows/system32** вы найдете внушительную коллекцию DLL-библиотек и исполняемых файлов Windows. Также в **~/.wine** есть файлы с настройками реестра (см. врезку «Реестр Windows», стр. 59). Все эти файлы вместе взятые образуют виртуальное окружение Windows. Во время работы *WINE* берет этот каталог из переменной окружения WINEPREFIX, и вы можете создать несколько разных Windows-окружений для разных приложений. За кулисами также есть серверный процесс *wineserver*, предоставляющий *WINE* ряд сервисов ядра, включая обмен сообщениями, доступ к реестру, отладку и частично управление окнами. Этот сервер запускается средой выполнения *WINE*, и вам беспокоиться о нем незачем.

> «WINE создает виртуальное окружение Windows — „бутыль".»

Для установки других программ Windows в теории достаточно найти их установочные пакеты и запустить их в *WINE*, примерно таким образом:

```
$ wine setup.exe
```

EPEL

EPEL (Extra Packages for Enterprise Linux — дополнительные пакеты для Enterprise Linux) — репозиторий дополнительных пакетов для Red Hat Enterprise Linux (или CentOS, или Scientific Linux), поддерживаемый Fedora Special Interest Group (http://fedoraproject.org/wiki/EPEL). Это пакеты из проекта Fedora, которые специально подбираются так, что они «никогда не будут конфликтовать или заменять пакеты базовых дистрибутивов Enterprise Linux». Чтобы воспользоваться EPEL, нужно включить репозиторий и загрузить ключ GPG.

В официальном репозитории CentOS есть маленький пакет под названием *epelrelease*, который сделат это за вас, и вы сможете включить EPEL обычной командой

```
# yum install epel-release
```

Я говорю «в теории», потому что на практике вы можете столкнуться с целым зоопарком сообщений об ошибках.

CodeWeavers идут на помощь

Если у вас не получается заставить приложения работать в *WINE*, рассмотрите *Crossover* от CodeWeavers. Это коммерчески поддерживаемая версия *WINE*, и команда CodeWeavers приложила массу усилий, чтобы пользоваться ею было проще. Она не бесплатная, но годовая подписка с бонусом в виде поддержки по телефону стоит всего £38 — это гораздо дешевле дня вашего времени, потраченного на то, чтобы заставить *WINE* работать; к тому же вам будет приятно знать, что вы вносите свой вклад в разработку *WINE*. Для образовательных учреждений цена чуть ниже, и есть 14-дневный пробный период, так что вам не придется расставаться с деньгами, не будучи уверенным, что приложение запустится. Лицензия выдается на пользователя, т. е. вы сможете установить *WINE* на несколько компьютеров, если в каждый момент времени будете пользоваться только одним из них. Как указано на сайте CodeWeavers, пойдете ли вы по бесплатному пути и будете пользоваться только *WINE* или предпочтете *Crossover*, зависит от вашего бюджета, технической компетенции и болевого порога.

На сайте *Crossover* тоже есть огромная база данных приложений с похожей схемой классификации — медальками золотой, серебряной, бронзовой, «не работает» и «не тестировалось». Она организована лучше, чем база штаба *WINE*. Всего я насчитал здесь 12700 приложений, из которых около 4500 — игры. Однако общее количество рабочих приложений гораздо меньше, так как очень много программ помечены как «не протестированные».

Crossover предоставляет вам графическое окно, откуда можно управлять своими приложениями Windows, и — что важнее — установщик, который *очень* упрощает этот процесс. Я решил опробовать *Crossover* в Centos 6.5. Процедура выглядит так:

» Включите репозитории EPEL в CentOS. (Как это сделать, поясняется во врезке «EPEL» внизу слева.)

» Зайдите на сайт www.codeweavers.com и нажмите кнопку Download Free Trial [Загрузить бесплатный пробник].

» Выберите свой дистрибутив Linux из выпадающего списка. (После этого будет определен нужный вам пакет.)

» Введите свой e-mail и нажмите кнопку Download. Загрузится файл с именем по типу **crossover-13.2.0-1.rpm**.

» Установите RPM с помощью *yum*:

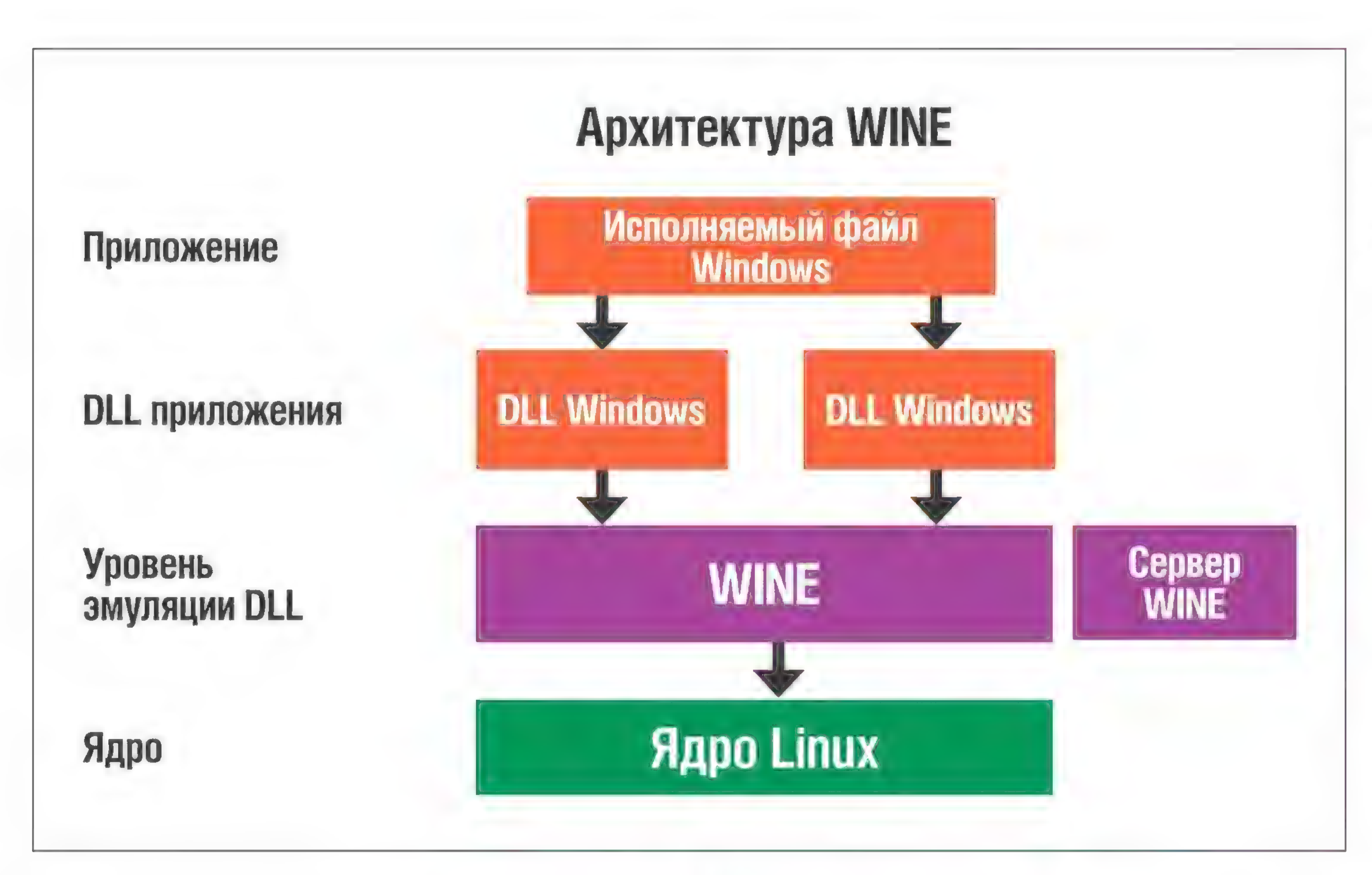
```
# yum install crossover
```

Этот пакет притянет за собой немало зависимостей (в моем случае — 62), но установка не должна вызвать проблем.

Crossover устанавливается в каталог **/opt/cxoffice** (и это одно из немногих на моей памяти приложений, которое действительно следует иерархии файловой системы и устанавливается в **/opt**). После этого *Crossover* также появится в меню Applications, и его можно запустить оттуда. В главном окне *Crossover* можно установить новые программы Windows, запустить уже установленными и управлять «бутылями» (которые обсуждаются ниже).

Лезьте в бутылку

Ранее я упоминал, что *WINE* создает виртуальное окружение Windows в каталоге **~/.wine**. В *Crossover* это окружение называется «бутылью», каковая идея развивается еще дальше, позволяя легко управлять несколькими бутылями. Такие бутылки могут выступать в качестве «песочницы», и приложения в отдельных «песочницах» не могут взаимодействовать друг с другом. С помощью бутылок также можно эмулировать несколько версий Windows



› *Wine* предоставляет слой совместимости, позволяющий запускать приложения Windows в Linux. Самые популярные из них — игры.

и переместить среду на другой компьютер. Бутыли *Crossover* хранятся в отдельных подкаталогах в **~/.cxoffice**, а не в **~/.wine** (изначально программа называлась *Crossover Office*).

Утилита управления бутылями *Crossover* позволяет просматривать их, создавать новые, просматривать в бутылях файловые системы, выводить список программ, установленных в них, изменять меню и файловые ассоциации и даже упаковать бутыль в файл .Deb или RPM, который легко установить на другой компьютер. И все это с комфортом графического интерфейса!

Пара пустяков

Простейший способ установки приложений Windows в *Crossover* — через функцию под названием "crosstie". По существу, crosstie — это набор действий (в виде XML-файла), который говорит установщику, как именно установить конкретное приложение. Если у приложения, которое нужно установить, есть файл crosstie, все должно пройти без сучка без задоринки.

Для примера установим бесплатную программу под названием *e-Sword* (для изучения Библии), которая доступна для Windows. Вот что нужно сделать:

» Зайдите на www.codeweaver.com и введите "e-Sword" в строке поиска. Результаты поиска покажут нам, что у *e-Sword* серебряная медаль.

» Щелкните по ссылке на страницу программы. Там можно просмотреть экранные снимки приложения.

» Нажмите на большую зеленую кнопку под названием Install e-Sword via CrossTie [Установить e-Sword через CrossTie]. Будет загружен файл crosstie (с именем **e-Sword.tie**), который по умолчанию откроется в установщике *Crossover*.

» Всю необходимую информацию установщик получит из файла crosstie, поэтому когда он запустится, просто нажмите на кнопку Install [Установить].

» Установщик создаст новую бутыль, где будет находиться приложение, а затем начнет установку.

» В моем случае *Crossover* решил, что мне также нужен *XML Parser* от Microsoft, и запустил соответствующий мастер установки.

» Далее просто следуйте окнам обычного мастера установки, как в Windows. По умолчанию программа установится в каталоге **C:\Program Files\e-Sword**, который, разумеется, находится в только что созданной нами бутылке.

» По завершении установки в главном окне *Crossover* появится кнопка запуска. Просто нажмите на нее — и вуаля! — программа заработает.

Установка с DVD

А вот другой пример. На сей раз мы установим *Microsoft Office* с официального DVD. Проясню: для этого не нужна лицензионная копия самой Windows, но нужна легальная копия устанавливаемой программы. Для установки, оказывается, достаточно воспользоваться файлом crosstie. Вставьте установочный диск в привод, запустите *Crossover* и нажмите на большую кнопку Install Windows Software... [Установить программу для Windows...]. Установщик автоматически найдет установочный DVD и программу установки на нем, а также автоматически заполнит параметры установки, поэтому мне оставалось только нажать Install. Опять же, установщик запустится и создаст новую бутылку, и опять же, окна мастера установки (включая ритуальную галочку «примите эту жуткую лицензию и забудьте о ней») будут такими же, как если бы вы устанавливали программу в настоящей ОС Windows. По завершении установки в главном окне *Crossover* появятся кнопки запуска для новых установленных компонентов.

Вот и все. И не забывайте: день без *WINE* — день беспросветный. Пока!

Реестр Windows

Одна из тех вещей, которые приходится эмулировать *WINE* — реестр Windows. Это иерархическая база данных, где хранятся настройки приложений Windows в форме ключей и их значений. Она по большей части заменила многие INI-файлы, применявшиеся в ранних версиях Windows. На реестр полагаются многие приложения, но (примечательно) .NET к ним не относится.
В *WINE* есть редактор реестра, очень похожий на своего собрата в Windows. Параметры реестра хранятся в **~/.wine** или в «бутылях» *Crossover*.

Рынок труда Linux

Зарабатываете на жизнь Linux? Поменяйте булочку из супермаркета на местные бриоши.

Жизнь без очков

Старому Ральфу было не по себе, когда его вежливо пригласили в комнату для интервью. Там он увидел министра по цифровизации Эда Мегабэнда [Megaband] и несколько джентльменов с военной выправкой.

«Пожалуйста, садитесь, Ральф. Мы просто хотим позадавать вам пару вопросов».

Ральф опустился на стул, не проронив ни слова.

«Итак, — Эд заглянул в свои заметки, — посмотрим. Назовите столицу Германии».

«Берлин!» — быстро и уверенно ответил Ральф. Эд поднял глаза на свой капюшон: там его ждал ответ.

«Наверное, у него стоит имплант или что-то в этом роде», — прошептал один из джентльменов-вояк. «Нет, — отвечал Эд. — Ни импланта, ни очков, ничего. Мы все проверили. Он просто... знает. Спросите что-нибудь другое».

Спрашивающий призадумался. «Когда Великобритания вышла из состава ЕС?»

Ральф сморщил лоб, но только на секунду. «Думаю, в 2018-м». Их капюшоны снова подтвердили его ответ.

«Откуда вы это знаете?» — резко спросил Эд. Правительство Эда беспокоила полная зависимость от поисковых машин. Они знали, что излучение всего одной электромагнитной бомбы, сброшенной врагом, сотрет с лица земли всю их цифровую инфраструктуру, и отчаянно искали место, где ее можно было бы сохранить. Подошел бы им Ральф?

«Я прочитал об этом в книгах, — сказал Ральф, не зная, чем все это закончится, — и просто... запомнил».

«Вы можете научить этому других?».

«Конечно».

Эд положил руку ему на плечо. «Тогда у нас есть для вас работа».

Рынок труда Linux растет. В отчете о рынке труда за 2014 год от Linux Foundation спрос назван «взрывным». Правда, в Foundation могли бы быть чуть предвзятыми, да и графиков, схем и таблиц в отчете не хватает. Тем не менее, приятно читать, что:

» «Более чем 9 из 10 специалистов по кадрам планируют нанять профессионалов в Linux в ближайшие шесть месяцев».

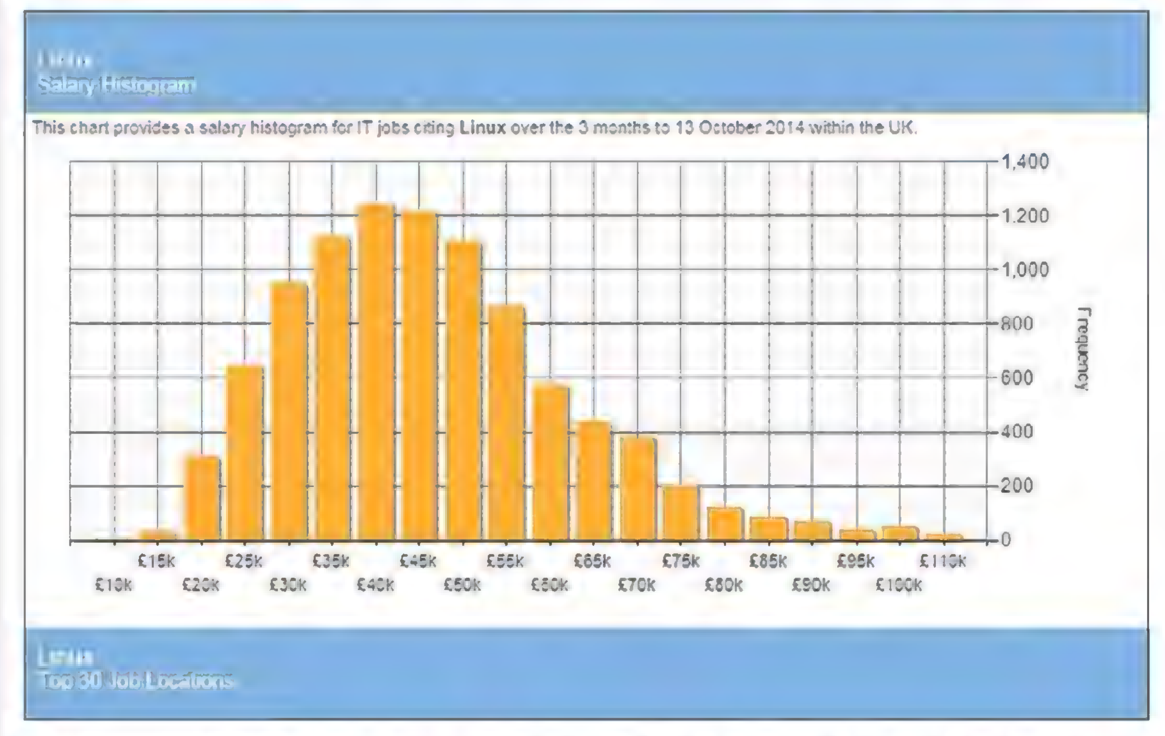
» «Профессионалы в Linux довольны ростом зарплат за последний год, который превысил средний рост зарплат для технических специалистов более чем на 2%».

» «86% респондентов говорят, что Linux положительно сказался на их карьере».

Весь отчет см. на http://bit.ly/LinuxJobs.

Итак, чего же достойны ваши знания Linux? На сайте www.itjobswatch.co.uk представлена различная статистика и тренды рабочих мест в сфере IT. Эти данные основаны на средних зарплатах, указанных в вакансиях в сфере IT в Великобритании со словом "Linux" в описании, размещенных за последние три месяца перед 13 октября 2014 года (т.е. их следует воспринимать не без известной доли предосторожности). На нашем графике показано распределение зарплат в Linux; в графике для ключевого слова "windows" зарплаты ниже примерно на £5000. Дневная ставка для подрядчиков по ключевому слову "linux" составляет £400, а для "Windows Server 2012" — £300. В некоторых областях рост впечатляющий — например, доля вакансий со словом "openstack" удвоилась (с 0,5% до 1,0%) за шесть месяцев с апреля по октябрь 2014 года. Hadoop также демонстрирует уверенный рост. (Но, пожалуйста, зайдите на сайт, найдите вакансии и регионы, которые интересуют вас, и сделайте собственные выводы.)

Несколько интересных сравнений есть на www.payscale.com. Средние зарплаты по слову "Linux" — у старших инженеров [Senior Software Engineers] — £46000, а у руководителей IT-отдела [IT managers] — £43000, тогда как у web-разработчиков — всего £20000. (Подробности см. на http://bit.ly/LinuxSalary.) Разумеется, зарплата — отнюдь не единственный критерий. Другие преимущества включают свободу работать дома, бонусы, возможности для обучения и интеллектуальное качество задач проекта, над которым вы работаете.



› **Распределение зарплат в вакансиях по ключевому слову "linux" неплохо выглядит на фоне зарплат в других сферах.**

Это не считается

Из того маленького уголка мира учебных курсов для IT, где я обитаю, всё большее принятие Linux не отражается на росте бизнеса, связанного с обучением. На прошлой неделе я читал «Введение в Linux» от имени крупного поставщика услуг обучения, и в классе было всего 8 студентов. Прямо по соседству находился класс по C#/.NET («двойной»), в котором было более 40 студентов. Считайте сами!

Знаете Linux? Докажите!

Сертификация и сопутствующее обучение — большой бизнес. Присоединяйтесь ко мне в прогулке по некоторым сертификационным программам Linux.

Итак, вы отправились на свое первое собеседование о Linux. Вы уверены в своих знаниях и слышали, что рынок труда Linux растет. Но как вы докажете потенциальному работодателю, что знаете Linux? У работодателя — похожая проблема. Скорее всего, у него просто нет никого, кто сможет задать хитрый вопрос: «Что делает бит sticky для каталога?» И в любом случае, докажет верный ответ только то, что вам знакомы хитрые вопросы о Linux. Так как же им убедиться в ваших способностях? Один из вариантов — сертификация для Linux, и на этом рынке есть несколько игроков. Здесь я рассмотрю троих: Linux Professional Institute, Red Hat и The Linux Foundation.

Linux Professional Institute

Начнем с Linux Professional Institute (LPI), который находится на рынке сертификации Linux с 2000 года. LPI предлагает довольно бесхитростный (и относительно свежий) курс для начинающих под названием 'Linux Essentials', за которым следует сертификация на трех уровнях. Цитирую их сайт (www.lpi.org/linux-certifications):

» «LPIC-1 — младший уровень сертификации администраторов Linux. Вы должны уметь выполнять задачи по техническому обслуживанию из командной строки, установить и настроить рабочую станцию и настроить простую сеть».

» «LPIC-2 нацелен на продвинутых профессионалов в Linux. Для получения этого уровня вы должны уметь администрировать смешанные сети малого и среднего размера и делать предложения по управлению сетью на более высоком уровне».

» «LPIC-3 предназначен для опытных профессионалов в Linux в промышленной среде. Вы должны уметь планировать и устанавливать LDAP-каталог, устранять неполадки, а также интегрировать его с Active Directory».

Углубитесь в любой из этих уровней, и вы найдете очень подробный список задач, включая список конкретных команд и файлов, которые должен знать обучающийся. У каждой темы есть «вес», который даст представление о том, сколько вопросов об этой теме вам будет задано.

Экзамены LPI по большей части не привязаны к дистрибутивам. (Ныне это становится сложнее, при таком количестве различий между дистрибутивами — *upstart* против *systemd*, *Yum* против *apt-get*, *SELinux* против *AppArmor*, KDE против Gnome и т.д.)

Вопросы на экзаменах LPI могут быть с вариантами ответов или с предложением «заполнить пустое поле». Экзамен можно сдать в Интернете в любом центре тестирования Pearson VUE, которых очень много по всему миру, поэтому где-нибудь поблизости обязательно один да найдется; чтобы его найти, просто зайдите на www.pearsonvue.com/vtclocator. Вам придется зарегистрироваться для получения идентификатора LPI (бесплатно), заранее записаться на экзамен и затем просто прийти. Стоимость экзамена — около $180 (и помните, что на каждый сертификат необходимо сдать два экзамена). Время от времени LPI также предлагает сертификацию на конференциях по Linux, часто со скидкой.

Тесты, основанные на простом выборе вариантов и заполнении полей, легко очернить, сказав, что они не позволяют оценить навыки решения проблем в реальном мире. Есть сильный соблазн сосредоточиться только на проверке названий и параметров низкоуровневых команд, потому что такие вопросы написать проще всего (поверьте, я сам этим занимался). Однако если тщательно подойти к составлению вопросов, то довольно легко понять, кто знаком с темой, а кто нет. Большое преимущество таких тестов, разумеется, в том, что их легко и дешево проверять и оценивать. Если вам нужно лишь пройти экзамен, а не глубоко изучить тему, то на некоторых сайтах есть «мозговые дампы» или (как утверждается) настоящие вопросы с этого экзамена — но этого я, конечно, не советовал бы.



› Пришел, сдал, получил сертификат!

Практическая сертификация

На другом конце спектра — сертификация Red Hat с практическими тестами, в которых вам нужно настроить на реальном компьютере (ну или на виртуальной машине) определенные сервисы определенным образом или исправить ошибки конфигурации, намеренно сделанные на этом компьютере. Так как на этом компьютере скорее всего есть man-страницы, то никого не волнует, помните ли вы, например, параметры командной строки для рекурсивного вызова **chmod**; вы сможете просто их посмотреть, когда понадобится. Вместо этого вы сосредотачиваетесь на собственно настройке системы и решении проблем.

Red Hat предлагает огромное количество вариантов сертификации. Основные (в порядке возрастания сложности) таковы: »

Мертв по поступлении

Были и другие программы сертификации, которые со временем скончались. В Canonical, например, был «Сертифицированный профессионал Ubuntu», по двум экзаменам LPIC-1 и еще одному экзамену в стиле LPI для Ubuntu. Я какое-то время активно там участвовал, будучи их «главным преподавателем» и составляя учебные материалы, но вскоре всё это свернули, потому что (как я подозреваю) в Canonical сделали вывод, что не смогут извлечь из данной программы прибыли.

› **Red Hat продолжает доминировать на рынке труда Linux, но и CentOS не слишком отстает.**



» Сертифицированный системный администратор Red Hat (RHCSA) (Раньше это называлось «Сертифицированный инженер Red Hat»)
» Сертифицированный инженер Red Hat (RHCE)
» Сертифицированный архитектор Red Hat (RHCA)

Есть и несколько более специализированных:

» Сертифицированный администратор JBoss Red Hat
» Сертифицированный администратор виртуализации Red Hat
» Сертифицированный системный администратор Red Hat OpenStack

Сертификация для OpenStack сейчас выглядит особенно интересной, учитывая степень распространения OpenStack, и у Red Hat сейчас есть особое предложение: при покупке сертификационного экзамена (EX210) вы бесплатно получите курс OpenStack Online (CL210R), хотя учитывая задержки, неизбежные при публикации бумажного журнала, это предложение может стать неактуальным, когда вы будете это читать.

Для получения RHCSA и RHCE сдается один экзамен (когда я сдавал экзамен RHCE, это заняло целый день). RHCA — другая история. Для его получения нужно сертифицироваться как RHCE и иметь по меньшей мере еще пять сертификатов. На данный момент доступны следующие:

Сертификация	Обучение	Экзамен
Развертывание и управление системами	RH401	EX401
Кластеры и хранилища	RH436	EX436
Настройка производительности	RH442	EX442
Укрепление сервера	RH413	EX413
Платформа как сервис	CL280	EX280
Виртуализация	JB450	EX450
Безопасность: сетевые сервисы	(отозван)	
Сервисы каталогов и аутентификация	(отозван)	
Политика администрирования SELinux	(отозван)	

Большинство учебных курсов в таблице длятся по 4 дня и стоят около £2000 каждый. Даже экзамены стоят по £600 каждый. Поэтому, по моим расчетам, переход с RHCE на RHCA обойдется вам в £3000 и еще порядка £13000, если вы решите пройти все рекомендованное Red Hat обучение. Как сказал бы Крис Террент [Chris Tarrant, известный телеведущий], это уже серьезные деньги. Возможно, это одна из причин, по которой количество специалистов с RHCA довольно мало. Поиск количества сертифицированных профессионалов Red Hat в Великобритании обнаружил 500 RHCSA, 470 RHCE и всего 14 RHCA. Конечно, это, возможно, и потому, что получить RHCA не так просто.

Red Hat — безусловный лидер на рынке сертификации Linux, и это дистрибутив номер 1, который интересует работодателей (см. график «тенденции рынка труда» слева). Но если вы пойдете по этому пути, будьте готовы получить сильно сдвинутый в сторону Red Hat взгляд на Linux и его администрирование.

Новые пацаны во дворе

Новые соседи по кварталу — в августе этого года на LinuxCon Linux Foundation анонсировала совершенно новую программу сертификации. Подробности — на http://training.linuxfoundation.org/certification. В программе предлагается два сертификата; для получения каждого из них нужно сдать всего один экзамен стоимостью $300. Цитирую их сайт:

» «Сертифицированный системный администратор Linux Foundation (LFCS) обладает знаниями по системному администрированию Linux на простом и среднем уровнях с командной строки».
» «Сертифицированный инженер Linux Foundation (LFCE) обладает более широким диапазоном и глубиной знаний по сравнению с сертифицированным системным администратором Linux Foundation (LFCS). Сертифицированные инженеры Linux Foundation отвечают за проектирование и реализацию системной архитектуры. Они предоставляют траекторию развития и служат „экспертами по теме [Subject Matter Experts]“ для следующего поколения системных администраторов».

Итак, эти сертификации в общем эквивалентны RHCSA и RHCE. Как и у Red Hat, экзамены практические: обучаемые выполняют задания и решают проблемы из командной строки. Но в отличие от Red Hat здесь можно выбрать, каким дистрибутивом воспользоваться — на данный момент это CentOS 6.4, openSUSE 13.1 или Ubuntu 14.01.

Экзамены Linux Foundation отличаются и тем, что сдать их можно где угодно — во всяком случае, из любого места с приличным интернет-соединением. Понадобится компьютер с браузером *Chrome*, но не придется устанавливать Linux или виртуальную машину — виртуальная машина будет запущена в браузере с помощью эмулятора терминала. У Linux Foundation есть утилита для проверки совместимости, с которой можно выяснить, удовлетворяет ли ваша система необходимым требованиям.

Так как эту схему можно надуть (простейший вариант — подкупив какого-нибудь гуру Linux, посадить его вместо себя), то есть специальный человек, который наблюдает за вами в течение всего экзамена — он получает звук и изображение с вашего рабочего места, а также может подключиться к монитору. Недавно я имел подобный опыт на учебных курсах и немного не уверен в том, как это будет работать у студентов при узкой полосе пропускания

Сертификация разработчиков

Всех предлагаемые сертификации Linux отличает упор на системное администрирование. Насколько я знаю, «Сертифицированных разработчиков ПО для Linux» нет. В W3schools есть несколько облегченных сертификатов для web-дисциплин вроде PHP, HTML5 и JavaScript. Cloudera предлагает «Сертифицированного разработчика для Apache Hadoop», «Сертифицированного аналитика данных» и еще два сертификата. Институт C++ предлагает сертификацию в C и заодно в C++ на профессиональном и опытном уровнях. На сайте http://bit.ly/prog_lang_certif есть краткий обзор (лучших семи, даже не лучших десяти) других программ сертификации для разработчиков. Экзамены во всех этих программах сдаются в Интернете, а вопросы предлагают лишь выбор ответа из нескольких вариантов, поэтому трудно предположить, насколько они котируются в профессиональной среде (если вообще котируются). Думаю, на месте кадрового специалиста вы обратите большее внимание на активное участие кандидата в проекте свободного ПО.

и большими задержками. А чтобы вас не заменил гуру, перед экзаменом вы должны предъявить свою фотографию.

Это необычный подход к сдаче экзамена, и будет интересно взглянуть, как это работает.

Сертификация и обучение

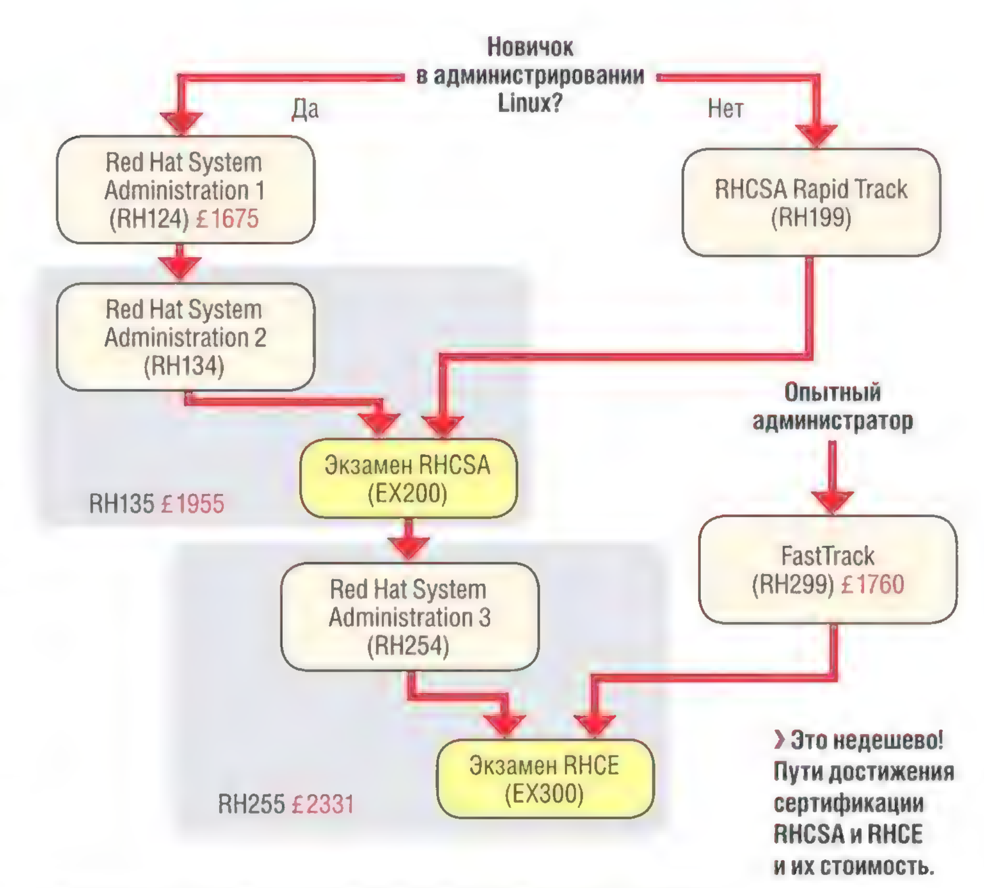
Сертификация, предлагаемая некоторыми компаниями, поддерживается учебными программами. Для некоторых компаний (не буду упоминать названия) сертификация — это просто способ заполнения учебных комнат. Обычно эти компании не разрешают вам сдать экзамены, если вы не посещали учебные курсы. С LPI все иначе; он вообще не предлагает учебных курсов и вполне удовлетворен тем, что вы просто сами пройдете обучение и сдадите экзамены. (Я всегда так и делал.)

При этом у LPI есть активная программа партнерства с поставщиками учебных курсов по всеми миру посредством программ ATP (Approved Training Partner — Утвержденный партнер по обучению) и AAP (Approved Academic Partner — Утвержденный академический партнер). Несколько лет LPI также настаивал, чтобы эти партнеры присылали свои учебные материалы на утверждение, но сейчас от этого требования отказались. Firebrand — один из почти десятка таких учебных партнеров в Великобритании. Их единственная задача — предоставить учебные курсы сертификационным компаниям (будь то LPI, Prince2, Microsoft MCSE или другие).

Вместо того, чтобы платить за учебный курс, вы можете пройти обучение по книге. Поиск книг на Amazon по запросу "Linux Certification" дает обильные результаты, хотя я читал только одну из них — "LPI Certification in a Nutshell [Сертификация LPI в двух словах]" издательства О'Рейли. Эта книга тесно привязана к темам экзамена LPI, но учтите, что она описывает только первый уровень LPI; к тому же последняя редакция книги вышла в 2010 году, и с тех пор темы экзамена изменились. (Это общая проблема бумажных изданий.)

У Red Hat есть активная программа обучения для ее активной сертификационной программы. На графике (справа) показаны лишь основные учебные курсы для сертификаций RHCSA и RHCE. Если вы последуете официальным путем и пройдете эти курсы (RH124, RH135 и RH255), то это займет три недели и будет стоить вам £5961 за полные курсы в учебных аудиториях. (Обучение в Интернете чуть дешевле; также можно купить учебные ваучеры, что сэкономит вам несколько процентов.) Одна из самых больших компаний, QA Training, предлагает полный набор курсов Red Hat на основе официального материала Red Hat и материалов независимых авторов. Учебные курсы опять же гораздо дешевле; я бы особенно порекомендовал книгу Асгхара Гхори [Asghar Ghori] "Red Hat Certified System Administrator and Engineer [Сертифицированный системный администратор и инженер Red Hat]", хотя она и написана для RHEL6, а не 7, и уже немного устарела.

Исторически обучение от Linux Foundation было сосредоточено на разработках и ядре, с такими курсами, как "Developing Linux Device Drivers [Разработка драйверов устройств для Linux]" и "Linux Kernel Internals [Внутреннее устройство ядра Linux]". Недавно они расширили свое портфолио курсами "Linux System Administration [Системное администрирование Linux]", "KVM Virtualisation [Виртуализация на KVM]", "OpenStack Cloud Architecture and Deployment [Разработка для OpenStack и его облачная структура]" и некоторыми другими.



› **Это недешево! Пути достижения сертификации RHCSA и RHCE и их стоимость.**

Должен признаться, что хотя я пробовал сдавать (и сдал) экзамены для сертификации RHCE, Novell CLP и LPI, но никогда не платил ни копейки за учебные курсы. Я тщательно изучал темы и затем просто пробовал делать то, что нужно. Установите *Apache*. Добавьте какие-нибудь файлы на сервер. Настройте управление доступом. Сломайте все. Исправьте ошибки. Повторите все сначала. Я обнаружил, что если записать свои команды и схемы процедур, то это огромное облегчение для памяти. Время, потраченное на обучение, заодно улучшило мои навыки по решению проблем — вас удивит, что способны натворить студенты на лабораторных занятиях...

Защита от подделок

Сертификат — это просто лист бумаги, и подделать его (или отсканировать действительный сертификат и изменить в нем имя кандидата) не так уж сложно. Для защиты от этого у Red Hat есть страница, где можно ввести номер сертификата и убедиться, что он принадлежит именно тому человеку, который его предъявил (http://bit.ly/RedHat_verify). На ней также можно найти лиц в заданном регионе, у которых есть нужные сертификаты (и связаться с ними по электронной почте). Сертификаты LPI также проверяемы (http://bit.ly/lpi_verify).

И последний комментарий. Лично я считаю, что сертификация играет свою роль для тех, кто пытается получить работу, связанную с Linux. Вместе с тем, нанимай я на работу гения в Linux, продемонстрированный им опыт администрирования Linux в реальном мире всегда имел бы преимущество над сертификацией. LXF

Военная байка

Несколько лет назад, будучи связан с учебной компанией, которая хотела стать учебным партнером SUSE Linux, я решил сдать «практический» экзамен Novell CLP. Вот смеху было! Это был практический экзамен, в котором следовало взаимодействовать с несколькими удаленными виртуальными машинами через удаленный рабочий стол в браузере. Экзамен протекал с вязкостью патоки; на каждое переключение из одной виртуальной машины в другую уходило не меньше минуты, а из-за проблемы с клавиатурой я не мог набрать один символ (кажется, символ канала), и его каждый раз приходилось копировать и вставлять из файла.

Как ни странно, экзамен мне все-таки удалось сдать.

LINUX FORMAT

Учебники

Наши эксперты помогут вам с любым приложением Linux!



ЕВГЕНИЙ БАЛДИН
Подтвердивший свою квалификацию физик.

FSF зажигает

Улыбаемся и машем.
Пингвины. «Мадагаскар»

В конце прошлого года FSF выпустило руководство Giving Guide 2014 — о том, как правильно выбирать подарки. Люди дарят подарки. Это нормально. Но люди часто упускают из виду ограничения, которые несёт с собой тот или иной выбор. В рамках своего руководства FSF рассказывает о некоторых свободных альтернативах, которые могут оказаться вполне к месту при выборе подарков. А раз уж так получилось, что определённая часть ваших денег была внезапно сэкономлена, то её можно потратить на поддержку свободных проектов. Ну или самого FSF — ведь, в конце концов, кто-то же должен быть правильно упёртым в этом несовершенном мире.

Готов признать, что некоторые позиции предвзятому читателю могут показаться излишне наивными. Ну как можно подарить *GIMP*? Очень просто: если есть талант к рисованию, то *GIMP*'а для самовыражения вполне себе достаточно. А чем плох подарок, сделанный с помощью свободных инструментов?

Да, свободное железо технически отстаёт от топовых предложений мира голого чистогана, зато в этом железе вы можете быть совершенно уверенными. Да, FSF тоже отнюдь не прочь насобирать денег (https://my.fsf.org/donate/) — куда от этого деваться? Но они по крайней мере не скучные.

E.M.Baldin@inp.nsk.su

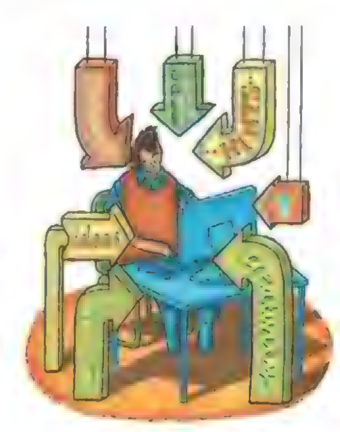
В этом месяце вы научитесь...

















И новичкам, и гуру! Всегда полезно будет познать нечто доселе неведомое

Journald: Логи для Systemd

Шон Конвэй покажет изнурённому сисадмину, как работать с *Journald* и вскрывать журналы новым инструментом, *Journaldctl*.

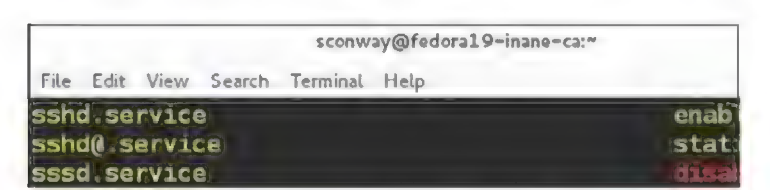


Наш эксперт

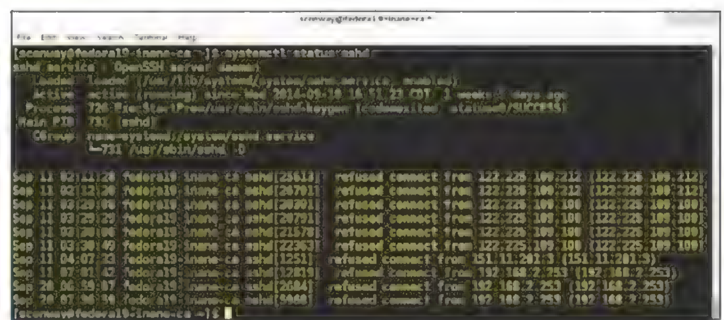
Шон Конвей — в прошлом специалист по авиационной электронике; теперь наслаждается жизнью, зная, что из-за него с неба уже не посыплются дюралевые обломки.



В 2 часа ночи у вас раздаётся отчаянный звонок от группы беспомощи (не опечатка) с сообщением, что компьютерный сервис недоступен. Вы входите в рабочую систему из дома, чтобы решить проблему удалённо, наскоро проверяете URL через браузер... и выясняется, что страница не открывается. Авторизовавшись на сервере, вы следуете привычной процедуре поиска неисправностей — просмотру файлов журнала. Зайдя в каталог, вы обнаруживаете, что файлы журналов всех процессов имеют размер 0. Это значит, что какое-то обновление ОС заменило старую систему журналирования другой. По новой системе



› **Команда list-unit-files в *Systemd* предоставляет информацию о статусе служб.**



› **Некоторые детали статуса службы и десять строк данных на одном экране. Вам понравится!**

файлы хранятся в другом месте, и для доступа к ним необходимы другие средства. Это маленькое изменение требует немедленного повышения квалификации, чтобы суметь понять проблемы системы.

Если вы мирно плыли по дорожке Fedora Linux, то вы, конечно, обнаружите, что Fedora 19 сделала *journald* логгером по умолчанию, заменив старую систему журналирования через *rsyslog*. При выходе Fedora 19 *Journald* страдал некоторыми детскими болезнями и в плане журналов был не лидером. Сейчас эти неполадки, к счастью, устранены, и в недавнем обновлении ПО *journald* принят как инструмент по умолчанию.

И всё же, зачем заменять средства журналирования? Да просто потому, что новый метод внедряет давно назревшие реформы. *Journald* улучшает журналирование так же, как *Systemd* оптимизирует процесс загрузки системы. Тем не менее, большинству системных администраторов из-за этого придётся расширять свои знания, выйдя из зоны комфорта.

Рекламировать читателям эти нововведения в наши цели не входит. Но они вносят в процесс журналирования некие стандарты, на которые в прежней системе тоже намекалось, но мало кто им следовал.

Поиск по журналам теперь осуществляется быстрее, благодаря индексации. Новая система производит двоичный вывод, не читаемый без особого инструмента. Доступ к журналам возможен только через него: обычными командами **cat**, **more** или **tail** этого не сделать.

Чтобы упростить процесс знакомства с инновациями, можно рассмотреть, как простые задачи решаются старыми методами и как — новыми. Понять принцип действия *journald* поможет инструмент *Journalctl*. Новые журналы имеют двоичный формат. *Journalctl* — это интерфейс командной строки (CLI), позволяющий просматривать их записи.

Раздобыть сведения об этом новом инструменте читатель может, введя код в CLI. В выводе будет представлен список флагов и опций доступных команд.

```
[sconway@fedora19]$ journalctl _COMM=sshd
-- Logs begin at Wed 2014-09-10 16:51:00 CDT, end at Mon 20
Sep 10 16:51:23 fedora19-inane-ca sshd[731]: Server listenin
Sep 10 16:51:23 fedora19-inane-ca sshd[731]: Server listenin
Sep 10 16:55:12 fedora19-inane-ca sshd[9177]: refused connec
Sep 10 19:30:56 fedora19-inane-ca sshd[5619]: refused connec
```

› **Файл журнала, аналогичный /var/log/secure в старой системе.**

```
journalctl -h
```

Просмотрев man-страницы *journalctl* с помощью

```
man journalctlt
```

вы получите расширенное описание с поддержкой запросов. Внизу страницы имеются ссылки на ресурсы. Пройдя по ссылке www.freedesktop.org/wiki/Software/systemd, вы найдёте лучший ресурс для поиска информации по *Systemd* и *Journald*. Примечание: URL указан в конце man-страниц, так что можно даже его не сохранять.

Если вы планируете глубоко погрузиться в *Systemd* и *Journald*, еще одним полезным источником знаний будет блог разработчика Red Hat, Леннарта Поттеринга [Lennart Pottering] (http://0pointer.net/blog/archives.html). Советуем прихватить с собой пару баллонов терпения, поскольку погрузиться предстоит глубоко и надолго. Большое спасибо Леннарту за такую помощь сообществу.

Разработчики снабдили инструмент автодополнением по клавише Tab, чтобы можно было видеть список опций для завершения команды. Это избавляет администратора от необходимости запоминать синтаксис команды.

```
journalctl <Tab> <Tab>
```

При добавлении опций к команде Tab будет продолжать добавлять атрибуты для запроса.

```
journalctl _COMM = <Tab> <Tab>
SSHD <enter>
```

Из приведённого примера подстановки следует, что *journalctl* имеет доступ к журналам, которые в прежней системе существовали в **/var/log/secure**.

Для выхода из *Journald* и *Systemd* нажмите c. Если вы выполняли команду наподобие **tail**, нужно нажать Ctrl+c, чтобы выйти из команды. Одни окна сообщают пользователю этот нюанс, другие — нет.

Прежде чем изучать прочие возможности *Journald*, сделаем остановку на *Systemd*, дабы просмотреть команды, связанные с журналированием. Небось некоторые из вас все еще расстроены внедрением *Systemd* вместо *SysV*.

```
systemctl list-unit-files
```

Скриншот интерфейса командной строки (на рис. внизу стр. 66) показывает, как работает служба sshd. Вопрос: есть ли в журналах какие-то записи о ней? Вы можете найти ответ с помощью

```
systemctl status sshd.service
```

Вы получили статус этой службы, а также файл из десяти строк (на рис. вверху стр. 66). Теперь попробуйте выполнить ту же команду с *Systemd*:

```
systemd-analyze blame
```

Не совсем в тему, но мы — как сороки, которых привлекают блестящие предметы: когда мы в ходе нашего исследования наткнулись на команду *Systemd*, отображающую время загрузки, мы даже бросили писать, чтобы выяснить, почему так медленно стартует служба *ntpdate*. Но вернемся к журналированию в *Systemd*...

Следующая команда отображает журналы службы *sshd*. При старой системе эта информация хранилась бы в защищенных файлах **/var/log/**:

Фундаментальные знания

Команды управления пользователями, доступа к системе, предоставления услуг, загрузки системы и системных команд являются образчиками фундаментальных знаний — тех знаний, на которых базируется всё остальное. Когда базовые знания меняются, системному администратору приходится, мягко говоря, трудно. Доступ к файлам журнала, сгенерированным приложениями или самой системой, является важнейшим ресурсом для поиска и устранения неисправностей. Поэтому знание о том, как система регистрирует информацию, также является базовым. Журналы предоставляют мельчайшие детали, способные помочь администратору в определении и, возможно, решении проблемы.

```
journalctl _COMM=sshd
```

При изучении записей возникло подозрение, что журналы зафиксировали какую-то вредоносную активность на порте поддержки службы *ssh*. Как понять, какие журналы обрабатывает *journald*?

```
journalctl -F _SYSTEMD_UNIT
```

Судя по выводу команды (на рис. внизу), *sshd.service* — одна из служб, использующих *Journald* для журналирования. В прошлом, основной командой сисадминов при изучении журналов и просмотра записей, сделанных в режиме реального времени, была **tail -f / var/log/messages**. Не бойтесь, в *Journald* такая возможность тоже есть.

```
journalctl -f
```

Пока эта команда выполняется в одном окне терминала, в другом запустите демон времени **ntpd c ntpq -p**. Журналы отображают ошибку в режиме реального времени, инструмент запросов *ntp* доступен только в режиме суперпользователя. Мы успешно добавляли **-f** в конце некоторых других команд *Journalctl*, чтобы журналы следовали в порядке их создания. Интересно, было ли решение об использовании атрибута **-f** продиктовано желанием разработчиков *Journalctl* использовать знания системных администраторов о команде **tail**?

Объединяя вместе журналы из различных служб, инструмент журналирования обеспечивает определённую гибкость при фильтрации данных.

Напомним, однако, что журналы индексируются, что обеспечивает возможность создания запросов. Попробуйте эти примеры:

```
journalctl --since=yesterday
journalctl --since=2014-12-15 --until=»2014-12-16 23:59:59»
```

Используя файлы журнала в качестве отправной точки для исследования проблем с компьютером, необходимо понимать принципы *Journald*, раз старая система подвела, оставив нулевую информацию. Вы увидите: основные команды достаточно просты, хотя недовольства некоторых системных администраторов это не развеет.

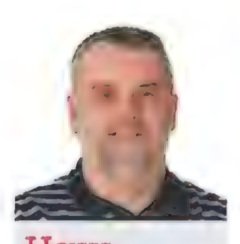
На данный момент, хоть уже больше 2-х часов утра, проблема решена, и мы отправляемся в будуар *Linux Format* за заслуженной кружкой теплого молочка и сладким сном. LXF

```
[sconway@fedora19]$ journalctl -F _SYSTEMD_UNIT
cups.service
ntpd.service
sshd.service
systemd-journald.service
```

› **Заметьте: при журналировании *sshd* показывается как служба.**

Rescatux: Чиним систему

Нейл Ботвик наступает на горло собственной песне и решает распространенные проблемы, не прибегая к командной строке.



Наш эксперт

У **Нейла Ботвика** по компьютеру в каждой комнате, но по соображениям безопасности он нипочем не расскажет, где находится центральный сервер.



Live CD с Linux — чудесное изобретение. С их помощью можно опробовать новые дистрибутивы, продемонстрировать Linux своим непросвещенным друзьям и починить сломавшуюся систему. Есть Live-дистрибутивы специально для восстановления неработающих систем, но их общий недостаток в том, что они требуют определенных навыков, а у большинства из нас система не выходит из строя так часто, чтобы эти навыки получить. Когда у вас сдох компьютер, и на экране — только мрачное сообщение от загрузчика, хочется побыстрее всё починить, а не загружаться с Live CD, чтобы открыть поисковик и добывать там корректные настройки *Grub* для данной ситуации.

Debian GNU/Linux 7.0.0 (wheezy) i386
Build: 2014-06-16 02:36:12
linux: 3.2.57-3+deb7u2
live-build: 3.0.5-1
live-boot: 3.0.1-1
live-config: 3.0.23-1
live-tools: 3.0.20-1
Boot menu
Live (amd64 failsafe)
Live (486)
Live (486 failsafe)
Install
Graphical install
Advanced options

› Rescatux работает в 32- и 64-битных системах. Пользоваться 32-битной версией в 64-битной системе обычно можно, но это не оптимально. Обратное невозможно.

По-моему, я знаю о загрузчиках не так уж мало, но даже у меня они не ломаются так часто, чтобы я с ходу исправлял бы ошибки из командной строки, ни разу не глянув в руководство с целью уточнить параметры (RTFM!).

Подготовка к операции

Нам нужен live-дистрибутив для решения общих проблем с системой без необходимости в специальных знаниях или исследованиях, и желательно такой, где незачем набирать длинные команды, опечатка в которых может усугубить катастрофу. Короче, такой, как Rescatux.

Rescatux загружается в облегченный рабочий стол *LXDE* как обычный Live CD, но окно, которое открывается при запуске, совсем другое. Это *Rescapp* — центр решения всех проблем. Не будем их перечислять, но рассмотрим те, что способны возникнуть, когда компьютер начинает безобразничать на нижнем уровне, и как Rescatux поможет их решить. Live CD, подобные Rescatux, не предназначены для борьбы с мелкими бедами пользователей: к ним обычно обращаются, когда произошло нечто серьезное.

При восстановлении системы часто приходится загружаться с Live CD — либо потому, что обычная загрузка не работает, либо для размонтирования корневой файловой системы. Также, как правило, нужны утилиты командной строки, и в Rescatux все это есть; но *Rescapp* во многих случаях значительно упрощает жизнь.

При нажатии в *Rescapp* любой из кнопок действий она срабатывает не сразу. Сначала открывается страница документации с объяснением данного действия, а поскольку многие действия касаются низкоуровневой природы, эту страницу стоит прочесть. Затем жмите кнопку Run! [Выполнить] в правом нижнем углу для выполнения операции.

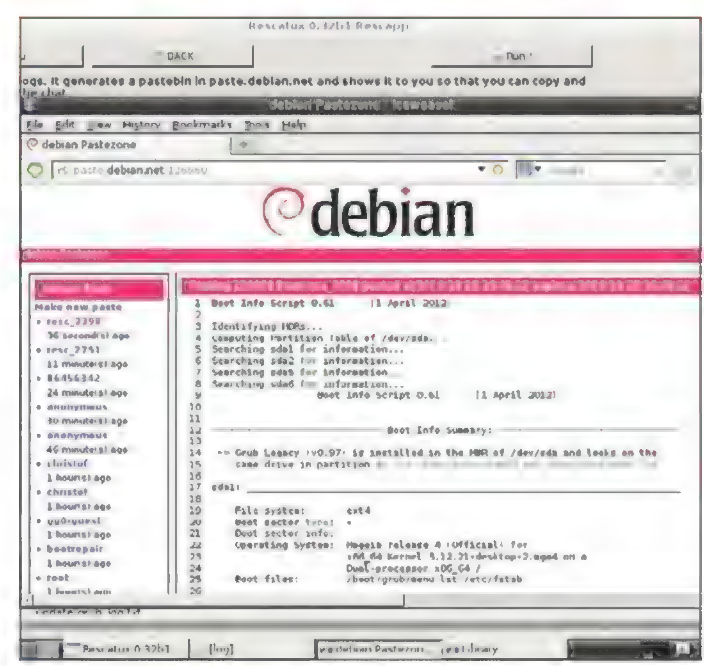
№1 Ошибки жесткого диска при загрузке

При возникновении ошибок файловой системы первым делом запускайте **fsck** (это сокращение от “filesystem check [проверка файловой системы]”, а не ругательство, которое вы издаете при виде ошибок). Это надо делать, пока файловая система размонтирована, для чего и нужен Live CD. Нажмите кнопку File System Check [Проверка файловой системы]. *Rescapp* на время смонтирует разделы, чтобы определить, какому дистрибутиву они принадлежат. Конечно, из-за повреждения диска разделы могут не смонтироваться (при загрузке дистрибутива часть небольших повреждений иногда исправляется незаметно для пользователя), и вы можете увидеть раздел с пометкой ‘Cannot mount [Невозможно смонтировать]’. Определить можно только корневые каталоги дистрибутива; если у вас отдельный раздел home, он будет отмечен как ‘Not detected [Не обнаружен]’ (или, если он поврежден, ‘Cannot mount

Дополнительная помощь

Хотя *Rescapp* может исправить массу проблем за несколько щелчков мыши, иногда нужна добавочная помощь. В *Rescapp* есть несколько полезных для этого утилит. Во-первых, web-браузер, в котором можно искать ответы; вводят туда сообщения об ошибках, которые вы получили при попытках восстановить систему. Всё, что делает *Rescapp*, записывается в журнал, и по нажатию на кнопку Show Log [Показать журнал] откроется список лог-файлов — помните, что это Live CD, и вы увидите список журналов только для текущего сеанса. Лог-файлы можно просмотреть и напрямую, они хранятся в каталоге **rescapp/logs**. Поиск в лог-файлах информации об операциях, которые вы пытаетесь выполнить, может дать вам полезную информацию. Если вас это не спасает, это может помочь другим — для этого есть кнопка Share logs [Поделиться лог-файлами]. После выбора лог-файла *Rescapp* отправит содержимое лог-файла в pastebin на серверах Debian, и вы получите URL. Скопируйте его в безопасное место, и сможете поделиться им с теми, кто захочет взглянуть на ваши лог-файлы. Для ответа в реальном времени попробуйте кнопку Chat. Откроется IRC-клиент с каналом **#rescatux**, куда можно скинуть URL своего pastebin и попросить о помощи. Разумеется, вы не ограничены этим каналом: можете поискать более конкретный совет в канале своего дистрибутива. Кнопка Share log on forum [Поделиться журналом на форуме] работает аналогично, позволяя вам попросить помощи на форумах **LXF** или на другом вашем любимом форуме.

Прежде чем звать на помощь в Интернете, воспользуйтесь кнопкой Boot Info Script [Скрипт с информацией о загрузке]. Она создает файл в каталоге **logs**, который содержит информацию о вашей системе и которым вы можете поделиться кнопкой Share log. Информация о вашей системе может иметь важнейшее значение для того, кто будет искать решение вашей проблемы.



› **Кнопка Share Log отправляет лог-файл в pastebin и дает вам URL, которым можно поделиться с тем, кто решит вашу проблему.**

[Невозможно смонтировать]'). Есть и другие причины, по которым раздел может не монтироваться; это может быть раздел подкачки или расширенный раздел, так что выбирайте разделы внимательно. В случае сомнений загляните в журнал Boot Info Script (о нем мы расскажем позже), где есть список разделов и их типы.

№ 2 Мой пароль не сработал!

Не считая тех ситуаций, когда загрузка заканчивается недружелюбной строкой grub>, это один из самых страшных моментов при работе с компьютером. Вы убедились, что набрали пароль правильно и что Caps Lock не включен. Возможно, вы забыли пароль или при установке новой системы допустили опечатку при его вводе.

Для сброса пароля потребуется загрузиться с Live CD и немного поиграть с **chroots** — просто изменить файл не получится — или воспользоваться *Rescapp*. Нажмите кнопку Change Gnu/Linux password [Изменить пароль Gnu/Linux] и, прочтя объяснение, нажмите Run!, выберите дистрибутив (их всегда будет как минимум два — ваш и Rescatux, который отображается как Debian 7) и пользователя, пароль которого нужно изменить. Введите новый пароль, и на этом все. На сей раз постарайтесь его не забыть! Эта кнопка подходит только для паролей Linux. Если у вас двойная загрузка с Windows, вы увидите отдельную кнопку для сброса пароля Windows.

№ 3 Я удалил не те файлы

И удивительно, и жутко то, какой ущерб может причинить простая опечатка. Например, вы хотели набрать

```
rm -f *.txt
```

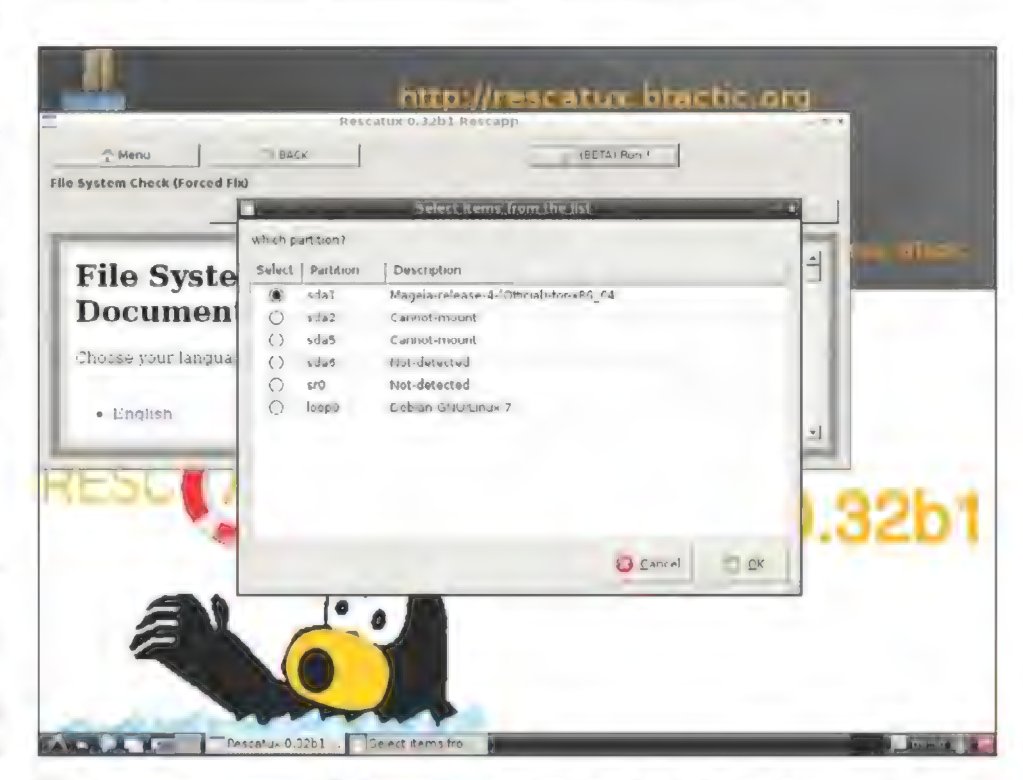
но вместо этого набрали

```
rm -f * .txt
```

Или хотели отформатировать **/dev/sdc**, но набрали **sdb**. Утилиты для восстановления файлов, такие как *extundelete*, конечно, существуют, но особо злостной опечаткой их легко одолеть; к тому же они не помогут, если разделы исчезли. Первое, что нужно сделать в такой ситуации — прекратить запись данных на диск, а если был поврежден раздел с корневой файловой системой, выключить компьютер командой:

```
sudo shutdown -n
```

Эта команда убивает процессы, не используя штатную систему **init**, что уменьшает количество записей на диск. Теперь загружайте Rescatux. Если диск был разбит на разделы, можно найти границы старых разделов командой **testdisk** и восстановить их. При переразбиении диска всего лишь производится запись в таблицу разделов; сами данные на дисках остаются нетронутыми, пока вы не отформатируете новые разделы. Поэтому если »

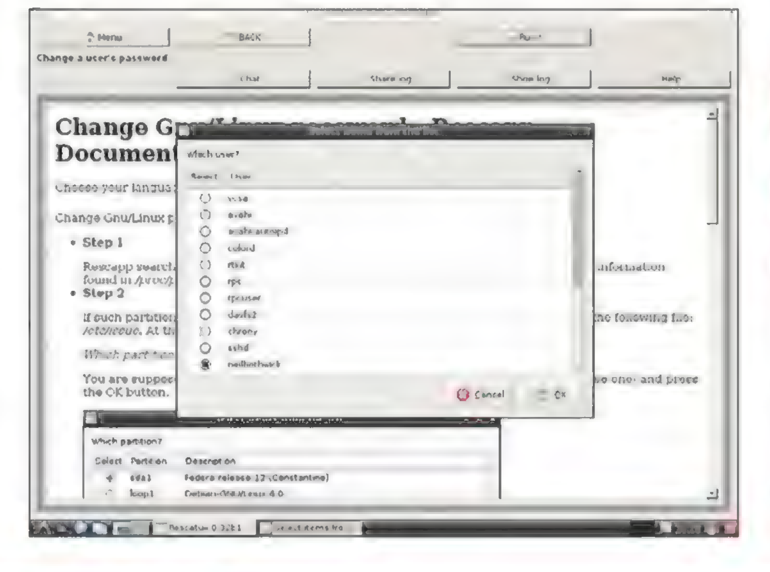


› **Если нужно восстановить файловую систему, выберите ее, а *Rescapp* доделает остальное. Сообщение "Cannot mount" может означать, что раздел поврежден, но здесь оно означает раздел подкачки и расширенные разделы.**

вы найдете границы старых разделов и вновь запишете их в таблицу разделов, то все должно вернуться в изначальное состояние. Что и делает **testdisk**. Подтвердив создание лог-файла, который может пригодиться нам позже, выберите сканируемый диск. В качестве типа раздела должны быть старомодные таблицы разделов MBR или EFI GPT для более нового варианта GPT; остальные варианты довольно специфичны. На съемных дисках иногда используется тип None, но обычно раздел с таблицей разделов MBR всего один. Выберите Analyse [Анализировать] для анализа таблицы разделов диска и затем Deeper Scan [Глубокое сканирование] для поиска утерянных разделов. Если вы нашли то, что искали, кнопка Write [Записать] должна вернуть утраченные настройки. Хотя об этом не во всех меню говорится, кнопка Q обычно возвращает вас в предыдущее меню. *Testdisk* — очень низкоуровневая утилита, и ее результат может быть необратимым, поэтому предварительно всегда делайте копию диска командой **dd**.

Если вы удалили файлы, а не разделы, то вам нужна утилита *PhotoRec*. *PhotoRec* сканирует диск на наличие следов файлов и пытается их воссоздать — для сохранения воссозданных файлов вам понадобится еще один диск. *PhotoRec* умеет находить только содержимое файлов: метаданные, такие как владелец, права доступа и даже имя файла, ей недоступны.

После восстановления у вас будет множество файлов с цифровыми названиями, хотя *PhotoRec* и выдаст им подходящие расширения на основании содержимого файлов. Если это цифровые фотографии (*PhotoRec* изначально восстанавливала файлы на стертых картах памяти) или музыка, то все тэги EXIF и ID3 должны сохраниться, что позволит сравнительно легко идентифицировать файлы. В противном случае придется потратить время и покопаться в файлах; но это лучше, чем полностью потерять данные.



› Забыли пароль? С *Rescapp* можно сбросить пароль любого пользователя Linux или Windows, в том числе root, всего несколькими щелчками мыши.

№4 У меня проблемы с Windows

Признайтесь, некоторые из вас пользуются Windows или имеют «друга», который ею пользуется. В *Rescapp* есть разнообразные возможности восстановления систем Windows, от сброса паролей до превращения пользователей в администраторов и других задач по управлению пользователями. В нем такж можно восстановить MBR Windows. Опция восстановления *Grub* подойдет только тем, у кого установлен хотя бы один дистрибутив Linux. Если вы хотите удалить с диска все разделы Linux, понадобится удалить *Grub* из загрузочного сектора диска и переустановить загрузчик Windows. В *Rescapp* это делается кнопкой Restore Windows MBR [Восстановить MBR Windows]. Выберите диск, как и в случае с восстановлением *Grub*, и на этом диске будет настроен загрузчик Windows.

№5 Это сложно

Пока мы говорили о решениях стандартных проблем, с которыми справляются несколькими щелчками мыши. Если все сложнее, то в Rescatux есть масса других программ, которые можно найти на дисках восстановления, и ими можно воспользоваться, просто открыв в терминале; но неудивительно, что есть и более сложные утилиты — в меню Expert Tools. К ним относятся:

» **GParted** — для (пере)разбиения жесткого диска.
» **Testdisk** — для поиска разделов и файловых систем на дисках с поврежденной таблицей разделов.
» **PhotoRec** — для восстановления удаленных или иным образом утраченных файлов с диска (не только фотографий).
» **OS Uninstaller** — для удаления лишних дистрибутивов из мультизагрузочной системы.

Стоит отметить, что при нажатии на кнопки Expert Tools сначала тоже открывается справочная страница, но это общая справка по Rescatux, а не по отдельным программам.

Кроме отдельных программ, есть кнопка Boot Repair [Восстановление загрузки], открывающая окно с параметрами для изменения процесса загрузки.

Загрузка ISO из Grub

Аварийные диски — это прекрасно, если вы сможете употребить их вовремя, то есть когда произойдет неприятность. Скопировать ISO-образ на USB-диск можно командой **dd**:

```
dd if=/lxfdvd/downloads/rescatux_cdrom_usb_hybrid_i386_amd64-486_0.32b1.iso of=/dev/sdX bs=4k
```

где **sdX** — ваш USB-диск. Так удобнее, но закон Мэрфи гласит, что в нужный момент вы этот диск не найдете, поэтому есть еще более удобный вариант. Чтобы вам не пришлось перебирать всю стопку DVD **LXF** в поисках диска с Rescatux, загрузим его с вашего жесткого диска.

Вам понадобится действующий *Grub2*, так что это вариант не на все случаи жизни; но если *Grub* и загрузочный раздел работают, так будет удобнее и быстрее. Скопируйте ISO-образ в каталог **/boot** и добавьте следующий текст в файл **/etc/grub.d/40_custom** (не меняя то, что там уже есть).

```
submenu "Rescatux 0.32" {
  set isofile=/Rescatux/rescatux_cdrom_usb_hybrid_i386_amd64-486_0.32b1.iso
  loopback loop $isofile
  menuentry "Rescatux 0.32 - 64 bit" {
    linux (loop)/live/vmlinuz1 findiso=$isofile boot=live config quiet splash
    initrd (loop)/live/initrd1.img
  }
  menuentry "Rescatux 0.32 - 32 bit" {
    linux (loop)/live/vmlinuz2 findiso=$isofile
    boot=live config quiet splash
    initrd (loop)/live/initrd2.img
  }
}
```

Теперь запустите **update-grub** или **grub-mkconfig** — или воспользуйтесь кнопкой Update Grub Menus в *Rescapp* для обновления меню. После перезагрузки системы у вас появится пункт для загрузки в Rescatux.

```
sudo grub-mkconfig -o /boot/grub/grub.cfg
```

Если при попытке загрузить его появится сообщение, что файл ISO не найден, добавьте следующую строку под строкой подменю

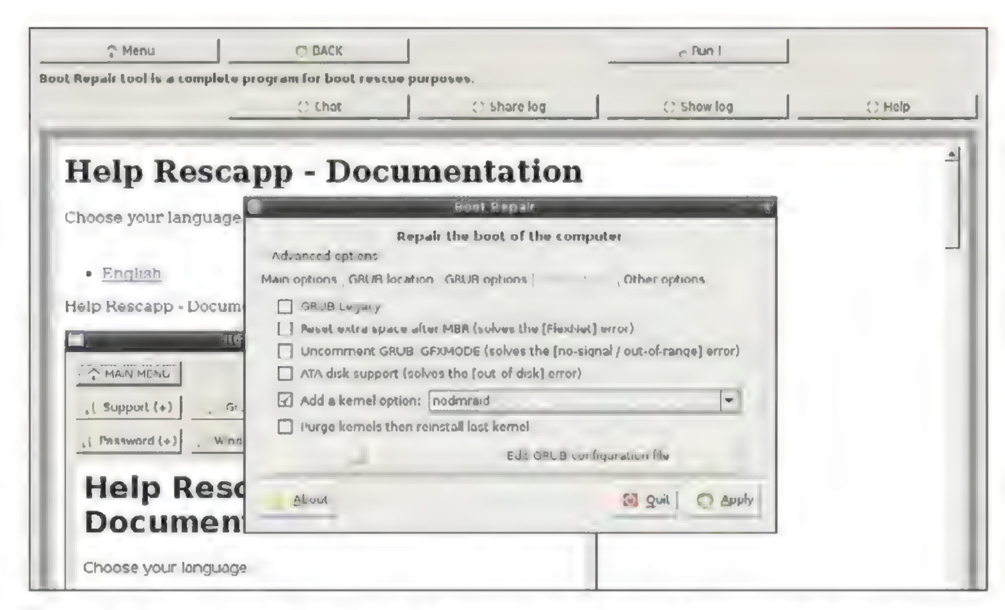
```
set root='(hd0,1)'
```

где **hd0,1** ссылается на первый раздел первого диска (по некой таинственной причине *Grub* отсчитывает диски, начиная с нуля, а разделы — с единицы). Измените нумерацию в соответствии со своей системой.

» **Пропустили номер?** Узнайте на с. 108, как получить его прямо сейчас.

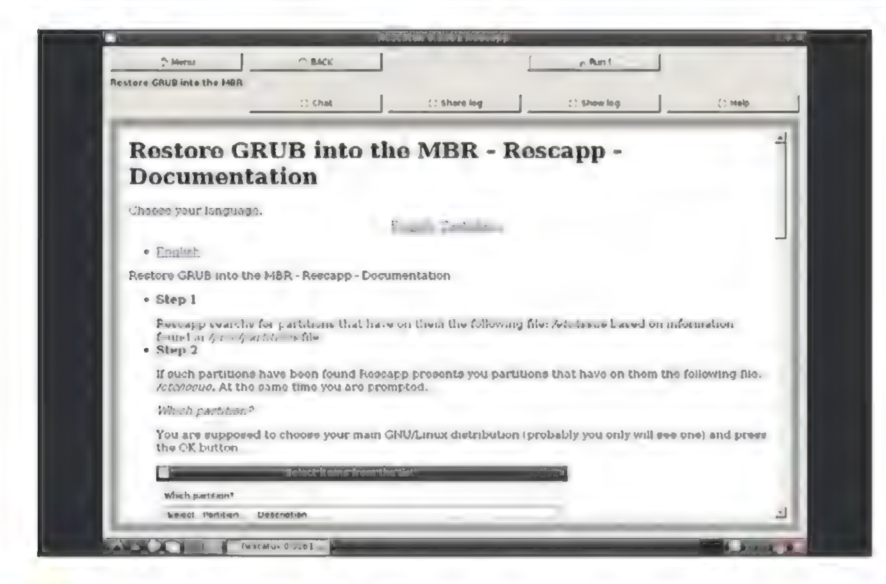
Это окно содержит несколько операций, особенно если вы поставили галочку Advanced options [Дополнительные параметры]. Оно позволяет сделать копию таблиц разделов и лог-файлов на USB-носитель — мудрый шаг, прежде чем начать что-либо делать. Большинство из приведенных здесь параметров позволяют изменить работу загрузчика. Для переустановки загрузчика есть отдельная кнопка, и можно также изменить его местоположение и параметры. Как часто решением вашей проблемы было всего лишь «добавить параметр xyz в Grub»! Для этого можно изменить файлы настройки, но в окне Boot Repair есть вкладка, где можно добавить параметры, не редактируя важнейшие системные файлы с присущим риском сделать все еще хуже.

На данный момент Rescatux находится в бета-версии и содержит кое-какие ошибки. Не работают одна-две опции, но ничего опасного мы не нашли. Опции скорее вообще не работают, а не делают что-то не так. Мы надеемся, что эти ошибки исправят в будущем, но Rescatux все равно стоит иметь под рукой, так как он избавит вас от головной боли во многих ситуациях. LXF



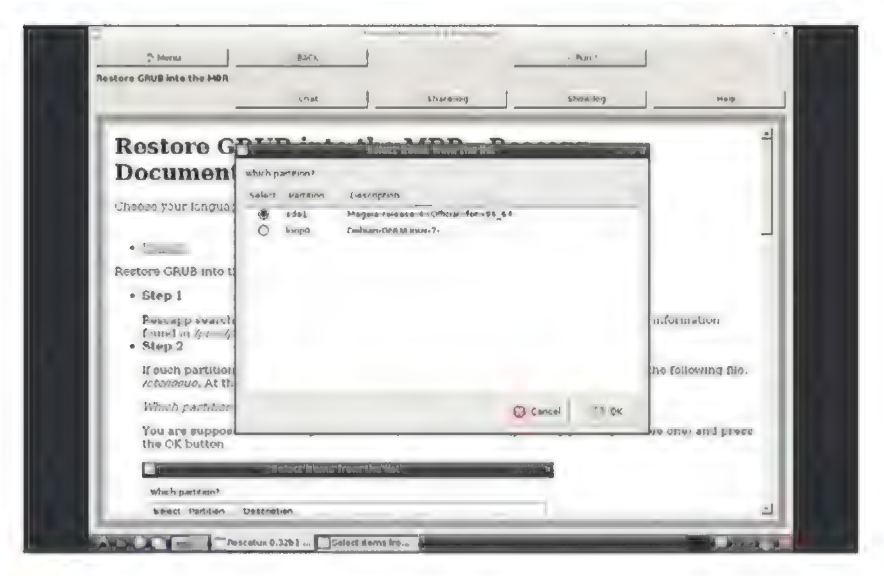
› Изменяйте параметры *Grub*, делайте резервные копии *Grub* и восстанавливайте их из меню Expert tools.

Излечиваем Grub



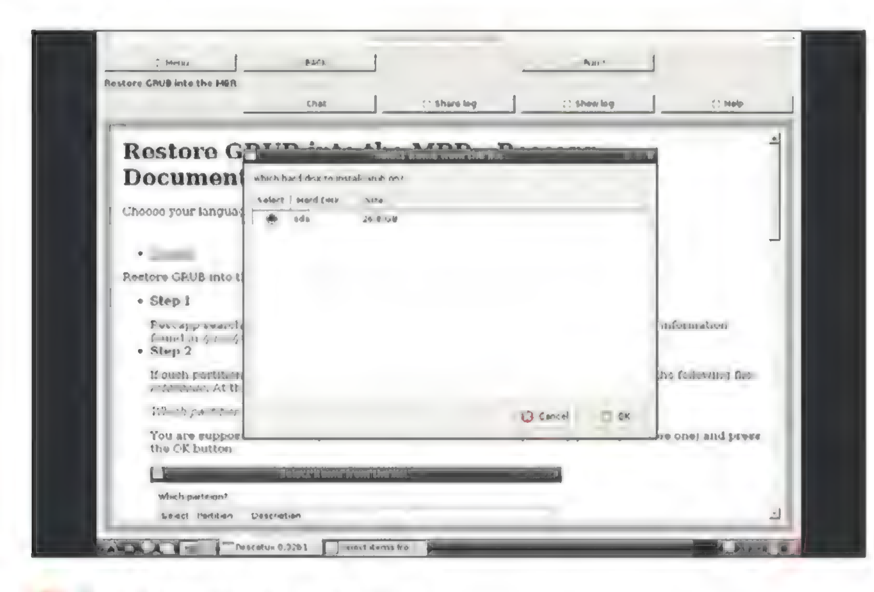
1 Восстановление Grub

Выберите Restore Grub [Восстановить Grub] в главном окне *Rescapp*, прочтите текст справки и убедитесь, что вы понимаете, что происходит. *Rescapp* сделает большую часть работы, но вам все равно нужно знать, на какой диск и с каким дистрибутивом должен использоваться *Grub*. Когда будете готовы, нажмите Run!.



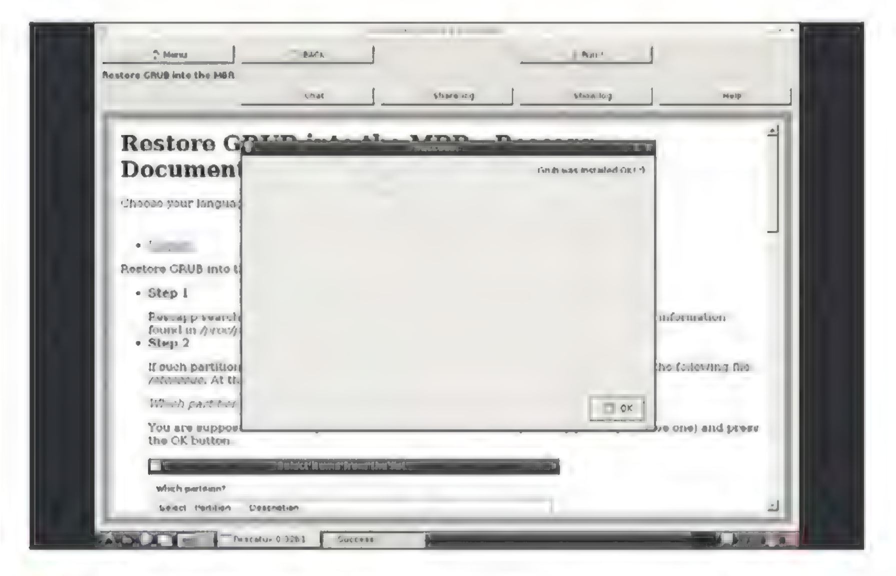
2 Выбор дистрибутива

Rescapp просканирует разделы диска на установленные дистрибутивы, попытавшись распознать их по имени, где возможно. Выберите один дистрибутив, который будет «основным». Это должен быть дистрибутив, который вы вряд ли захотите удалить, поскольку для этого придется повторить весь данный процесс.



3 Выбор диска для Grub

Теперь выберите диск, на который хотите установить *Grub*; обычно это sda, то есть тот же диск, где расположен основной дистрибутив, хотя и не обязательно. Если вы загрузились с флэшки, она может распознаться как sda, а ваш основной жесткий диск — как sdb.



4 Автоматическое исправление

Нажмите OK, и *Rescapp* выполнит необходимые действия, смонтировав и размонтировав необходимые файловые системы и запустив **grub-install** с нужными аргументами **--boot-directory**, **--root-directory** и **device**. Теперь можете загрузиться! Вот так все просто.

Screen: Множим терминалы

Нейл Ботвик использует программу со скромным названием *Screen*, чтобы выжать больше из терминала и оболочки — не перескакивая со стула на стул!



Наш эксперт

У **Нейла Ботвика** по компьютеру в каждой комнате, но по соображениям безопасности он нипочем не расскажет, где находится центральный сервер.

Screen — это мультиплексор терминала, и этим всё сказано, если вы уже знаете, что это такое. Но лучше описать его так: *Screen* — одна из самых полезных программ для пользователей терминала, особенно для тех, кто работает с удалёнными оболочками вроде SSH. С его помощью можно запускать один или несколько отдельных сеансов внутри одного терминала или виртуальной консоли. В некоторых графических терминалах есть вкладки, что позволяет запускать несколько терминалов в одном окне, но *Screen* умеет гораздо больше.

Давайте это проверим. Вероятно, *Screen* у вас уже установлен; в противном случае вы найдете его в менеджере пакетов вашего дистрибутива — это стандартная программа GNU. Откройте терминал и наберите

```
screen
```

Окно терминала очистится, но в остальном как будто ничего не произойдет. А на самом деле, вы работаете в новой оболочке внутри экрана. Введите команду, которая выполняет вывод, скажем **ls**, а затем нажмите Ctrl+A C. Вывод исчезнет, а в нижней части терминала появится строка состояния. Вы только что открыли в *Screen* второе окно. Теперь нажмите Ctrl+A N, и увидите вывод из вашей предыдущей команды. Поясним, что здесь происходит. Ctrl+A — это символ команд *Screen*: его использование сообщает программе, что нажатие следующей клавиши будет командой для выполнения. Так, C создает новое окно экрана, N переходит в следующее открытое окно (а P — в предыдущее). Если открыто несколько окон, все они будут отображаться в строке состояния в нижней части терминала, каждое со своим номером и именем процесса, запущенного в нём — чтобы перейти к конкретному окну, нажмите CTRL+A и его номер. Полный список команд *Screen* можно посмотреть, нажав Ctrl+A ?.

› **Если вы запутались, нажмите Ctrl+A ?, чтобы просмотреть список команд *Screen*.**

```
                         Screen key bindings, page 1 of 2.

                         Command key:  ^B   Literal ^B:  B

break       b         info        i          other       ^B        time      ^T t
clear       C         kill        K          pow_detach  D         title     A
colon       :         lastmsg     ^M m       prev        ^P p ^?   vbell     ^G
copy        ^[ [      license     ,          readbuf     <         version   v
detach      ^D d      lockscreen  ^X x       redisplay   ^L l      width     W
digraph     ^V        log         H          remove      X         windows   ^W w
displays    *         login       L          reset       Z         wrap      ^R r
fit         F         meta        B          screen      ^C c      writebuf  >
flow        ^F f      monitor     M          select      '         xoff      ^S s
focus       ^I        next        ^@ ^N sp n silence     _         xon       ^Q q
help        ?         number      N          split       S
history     { }       only        Q          suspend     ^Z z

                    [Press Space for next page; Return to end.]

                                        @hactar  Sunday 12 October  22:34
```

Туда и обратно

Пока что мы не увидели ничего такого, чего нельзя было бы сделать в X-терминале со вкладками, поэтому запустите программу, которой на выполнение понадобится некоторое время (например, *Sleep*, но какая-нибудь программа компиляции или перекодировки видео будет нагляднее), нажмите Ctrl+A D... и её нет. Теперь закройте терминал, откройте другой, или даже нажмите Ctrl+A+F1 для входа в виртуальную консоль, и выполните

```
screen -r
```

Вот и ваша программа — по-прежнему работает. Это полезно: можно запустить длительный процесс, не держа при этом терминал открытым. Вы даже можете уйти из рабочего стола, а затем вновь подключиться через виртуальную консоль (или наоборот); но по-настоящему смысл этого проявляется в сочетании с SSH. Вы можете войти в удаленный компьютер, запустить процесс и снова выйти. Он будет продолжать работать, даже если потом вы войдёте с другого компьютера.

Учтите, что возобновление сеанса с помощью **screen -r** невозможно, если у вас одновременно запущены несколько сеансов *Screen* (а не окон в одном сеансе), поскольку программа не понимает, который именно возобновлять. Можно просмотреть сеансы с помощью

```
screen -ls
```

и возобновить нужный, указав **screen -r** его номер, например, так:

```
screen -r 985
```

Если надо следить за двумя процессами одновременно, то постоянно набирать Ctrl+A N для переключения между окнами бывает утомительно, но *Screen* умеет показывать и несколько окон в одном терминале. Нажмите Ctrl+Shift+S (именно заглавную S — команды *Screen* чувствительны к регистру), и вы увидите, что окно вашего терминала разделено по горизонтали на две части. Новая область будет пуста, поскольку окна за ней ещё не закреплено, так что нажмите Ctrl+A, затем N, C или номер окна, чтобы его отобразить. Теперь у вас есть две области; каждая отображается отдельно. Вы можете делить дальше, или использовать Ctrl+A | и разбить область по вертикали — это поможет лучше организовать доступное пространство. В качестве примера, в одной области можно просматривать файл журнала с помощью **tail -f**, а в другой выполнять команду, хотя есть и множество других применений, особенно если вы работаете сразу с несколькими компьютерами через SSH.

В любой области используются обычные клавиши, и есть ещё несколько комбинаций для управления областями. Ctrl+A Tab переключает на следующую область, а Ctrl+A Shift+Tab — в обратном направлении. Ctrl+A Shift+X удаляет текущую область, а Ctrl+A Shift+Q — все прочие. Поскольку области являются частью одного и того же сеанса *Screen*, при удалении одной из них перестаёт отображаться только её содержимое; а окно *Screen*

```
Oct 12 22:39:14 hactar sudo:     nelz : TTY=pts/10 ; PWD=/home/nelz
-f /var/log/messages
Oct 12 22:39:14 hactar sudo: pam_unix(sudo:session): session opened
Oct 12 22:39:22 hactar sudo: pam_unix(sudo:session): session closed
Oct 12 22:39:49 hactar sudo:     nelz : TTY=pts/10 ; PWD=/home/nelz
-f /var/log/messages
Oct 12 22:39:49 hactar sudo: pam_unix(sudo:session): session opened
Oct 12 22:40:01 hactar cron[29760]: (root) CMD (test -x /usr/sbin/ru
Oct 12 22:40:38 hactar sudo:     nelz : TTY=pts/9 ; PWD=/home/nelz ;
obe -v sit2fe
Oct 12 22:40:38 hactar sudo: pam_unix(sudo:session): session opened
Oct 12 22:40:38 hactar sudo: pam_unix(sudo:session): session closed

[nelz@hactar ~ 0]%
[nelz@hactar ~ 0]%
[nelz@hactar ~ 0]% sudo modprobe -v sit2fe
modprobe: FATAL: Module sit2fe not found.
[nelz@hactar ~ 1]%
```

› Возможность разделить экран позволяет просматривать журнал во время выполнения команды или следить за выводом любого числа одновременных процессов.

остаётся на месте, и его можно просмотреть в другой области. Как завершить сеанс *Screen*? Это происходит при закрытии последнего окна, а выходите вы так же, как и из любого сеанса оболочки, нажав Ctrl+D.

Настройка экрана

При запуске *Screen* пытается прочитать два файла настройки: **~/etc/.screenrc** содержит глобальные настройки, **~/.screenrc**, если он есть, содержит параметры для вашего пользователя, которые имеют приоритет над глобальными параметрами. Настройки поясняются на man-странице, но те, что стоят по умолчанию, как правило, устраивают. Менее опытные пользователи *Screen* часто попадаются на том, что Ctrl+A используется не только в нём, но также и в терминале, как команда «перейти в начало строки». Во втором случае, Ctrl+A нужно нажать дважды, или же вы можете изменить командную клавишу на нечто неконфликтное. Я предпочитаю Ctrl+B (так же, как по умолчанию в *tmux*) и меняю это с помощью такой строки:

```
escape ^bB
```

Еще одно изменение, которое я обычно вношу — перманентное включение заголовка в нижней части терминала. По умолчанию он проявляется только тогда, когда у вас открыто более одного окна, потому сеанс с одним окном становится неотличим от обыкновенного терминала. Отображаемые в подписи данные также могут быть настроены — вот мои настройки по умолчанию:

```
caption always "%{= kw}%-w%{= BW}%n %t%{-}%+w %-=@%H - %LD %d %LM - %c"
```

Здесь "caption always" обозначает обязательное включение заголовка, а опциональная строка после него указывает, какая информация будет в нём отображена. Значения строк формата вы можете прочитать в разделе "string escapes" в man- или info-страницах *Screen*.

Присоединить и отсоединить

Вы можете вернуться к отсоединённому сеансу, используя **screen -r**, и, возможно, с помощью ID, но также и присоединиться к сеансу, который уже используется. Параметр **-d** велит *Screen* отсоединить сеанс.

Если присоединённых сеансов нет, использовать **-r** не удастся, как и если есть более чем один.

```
screen -R -d
```

подходит в первом случае, присоединяясь к открытом сеансу или создавая новый. При использовании **-RR** вместо **-R** *Screen* справится с несколькими сеансами, молчаливо подключившись к первому из них.

Попробуйте tmux

Screen — не единственный доступный вам мультиплексор терминала, хотя и наиболее широко используемый и рекомендуемый. Одним из вариантов является *tmux* (на рис. внизу), примерно с теми же функциями, но работающий немного по-другому (*tmux* использует модель клиент–сервер). По-моему, у него также лучше настройки по умолчанию, хотя в обеих программах их настолько легко изменить, что особой роли это не играет. Можете, конечно, установить и *Screen*, и *tmux*, и переключаться между ними, пока вы не решите, какой вам больше по нутру.

Больше, чем оболочка

До сих пор мы запускали *Screen* только чтобы попасть в сеанс оболочки. По умолчанию он запускает ту, что вы предпочитаете (как правило, *Bash*, разве что вы, как подлинный ценитель, перешли на *Zsh*), и предоставляет вам командную строку, из которой можно запускать программы.

Тем не менее, вы можете сделать так, чтобы *Screen* при старте запускал программу:

```
screen --screen-options command ---command-options
```

Запустится сеанс *Screen*, где выполняется ваша команда, а затем вы можете отсоединить её, когда захотите. По завершении выполнения команды сеанс *Screen* также будет завершён. Вы можете изменить это с помощью команды **zombie** в **/etc/.screenrc** или **~/.screenrc**. Если раскомментировать эту строку, окна будут оставаться открытыми до нажатия определённой клавиши (по умолчанию ^{). Если вы укажете две клавиши, первая будет осуществлять выход из окна, а вторая — повторный запуск команды. Вы также можете вставить после клавиш **oneerror**; в этом случае окно экрана остается открытым, только если команда завершилась с ошибкой — полезная настройка по умолчанию.

Желая использовать экран для запуска команды из сценария запуска, наберите

```
screen -d -m command
```

чтобы создать сеанс *Screen*, выполните команду и отсоединитесь от неё, чтобы процесс выполнялся в фоновом режиме.

Хотя поначалу *Screen* может показаться сложным, мало кто из пользователей терминала потом в силах от него отказаться, так что пробуйте! LXF

```
bind-key        ! break-pane
bind-key        " split-window
bind-key        # list-buffers
bind-key        $ command-prompt -I #S "rename-session '%%'"
bind-key        % split-window -h
bind-key        & confirm-before -p "kill-window #W? (y/n)" kill-window
bind-key        ' command-prompt -p index "select-window -t ':%%'"
bind-key        ( switch-client -p
bind-key        ) switch-client -n
bind-key        , command-prompt -I #W "rename-window '%%'"
bind-key        - delete-buffer
bind-key        . command-prompt "move-window -t '%%'"
bind-key        0 select-window -t :0
bind-key        1 select-window -t :1
bind-key        2 select-window -t :2
bind-key        3 select-window -t :3
bind-key        4 select-window -t :4
bind-key        5 select-window -t :5
bind-key        6 select-window -t :6
bind-key        7 select-window -t :7
bind-key        8 select-window -t :8
bind-key        9 select-window -t :9
bind-key        : command-prompt
[0] 0:[tmux]*                              nelz@hactar 22:35 12-Oc
```

› Это *tmux*. Делает почти то же, что и *Screen*, но немного иначе.

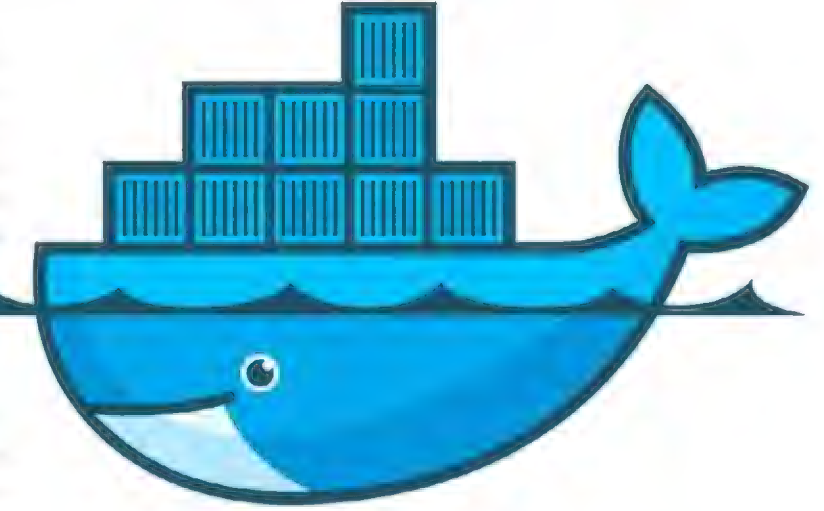
Docker: Jenkins и Dockerfiles

Джолион Браун рассматривает некоторые задачи, относящиеся к внедрению Docker в среду разработки, включая непрерывную интеграцию с Jenkins.



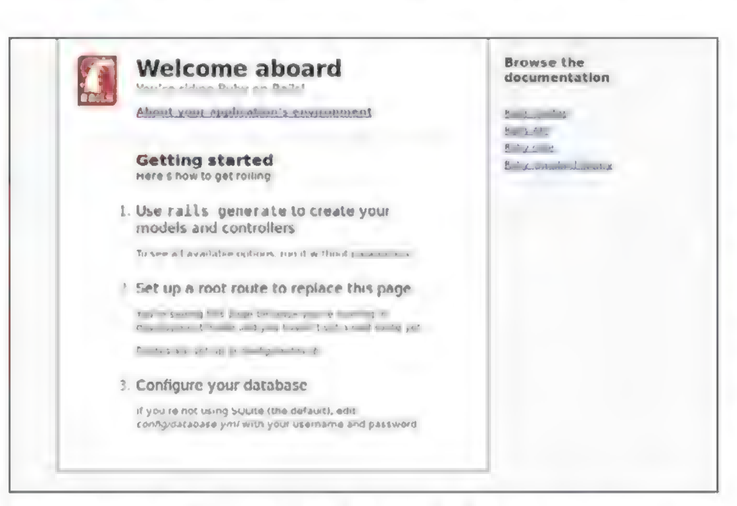
Наш эксперт

Джолион Браун
К ужасу жены, Джолион ушел со своей стабильной и уважаемой работы и начал карьеру фрилансера со специализацией на *DevOps* и *OpenStack*.

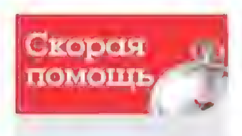


На уроке прошлого месяца [**LXF190**, стр. 78] мы представили Docker, реализацию программных контейнеров в Linux, и рассмотрели главные функции и команды. На основе этой работы мы рассмотрим некоторые шаги по внедрению Docker в среду разработки, разобравшись с возможностями распределенного доступа к контейнерам Docker среди членов команды и применением Docker в рабочем процессе непрерывной интеграции [continuous integration] (CI) с использованием хорошо известного инструмента Jenkins, а затем бегло глянем на некоторые задачи, с которыми обязан знакомиться системный администратор перед запуском любого сервиса: например, как делать резервные копии или записи лог-файлов, и т.п. Установку Docker мы заново описывать не будем; если вы не уверены в своих знаниях, просмотрите **LXF190** или простые инструкции на www.docker.com.

Обычно в компаниях, занимающихся ИТ-услугами, имеется несколько сред, где запускают одно и то же приложение на разных стадиях его разработки. Среда 'Dev' может быть, например, личным ноутбуком разработчика. Prod означает 'Production' [Производство] или 'Live'. Другие возможные среды — UAT (user acceptance testing — приемочное тестирование пользователя), DR (disaster recovery [аварийное восстановление]) или Pre-Prod (приближенная к условиям производства; применяется для исправления проблем эксплуатации). Разные версии приложения проходят по этим средам (надо надеяться, что линейно, но бывает, что и нет), пока не выйдут в производство. В старых, традиционных инфраструктурах каждая из этих сред могла состоять из одного или нескольких физических компьютеров, на которых запускается Linux с локального диска. Поддержка и эксплуатация таких серверов способна стать сущей головной болью для системного администратора. В идеале эти среды должны быть идентичны, чтобы размещенные в них приложения работали одинаково, и для обеспечения этого надо преодолевать разные препоны. Несмотря на все средства, имеющиеся в распоряжении сисадминов, до сих пор можно услышать знаменитое: «а на моем компьютере всё работало». Docker напрямую предназначен для решения этой проблемы, позволяя быстро собрать приложения из компонентов и заставляя один и тот же контейнер работать в любой нужной нам среде.



› **Rails, в Docker. Может, в это и не верится, но начальная страница Twitter когда-то выглядела именно так.**



Скорая помощь

Полное руководство по командам Dockerfile и лучшее практическое руководство находятся на http://docs.docker.com/reference/builder.

Польза Dockerfiles

На первом уроке мы для раскрутки контейнеров Docker активно использовали командную строку, но на практике большая часть разработки с Docker будет обходиться файлами Dockerfiles. У этих простых текстовых файлов есть то преимущество, что они легко поддаются контролю версий (их можно хранить в Git или в любом другом инструменте управления исходником), и хотя обычно они проще среднего скрипта оболочки, в плане создания систем они очень мощны. Вот пример одного, который выводит полный набор программ Ruby on Rails:

```
FROM phusion/passenger-ruby21:0.9.12
ENV HOME /root
CMD ["/sbin/my_init"]
RUN gem install rails
RUN cd $HOME; rails new lxf
RUN apt-get install git -y
RUN apt-get clean && rm -rf /var/lib/apt/lists/* /tmp/* /var/tmp/*
```

Чтобы протестировать этот Dockerfile, создайте его в новой (пустой) директории. В этой директории просто скомандуйте

```
sudo docker build -t railstest .
```

У вас скачается базовый образ из центра Docker — репозитория образов, поддерживаемого Docker Inc — и применит свои разнообразные способы, чтобы получить указанную версию (в нашем случае — 0.9.12). Passenger-ruby — это образ Docker,

созданный хорошо известной (по крайней мере, в кругах Ruby) командой разработки Phusion (Passenger — это web-сервер/сервер приложений, более всего известный за размещение приложений Ruby on Rails, но он умеет и массу другого). Мы добавили команды **gem install rails** и **cd $HOME; rails new lxf**. Все, кто недавно устанавливал Rails, знает, что это процесс весьма длительный и требует нескольких команд. Docker легко с этим справляется благодаря нашему повторному использованию образа Passenger (хотя первоначальное скачивание может отнять немало времени).

По завершении скачивания и установки мы можем запустить наш контейнер Docker следующим образом:

```
sudo docker run -p 3000:3000 —name lxf -t -i railstest /bin/bash
```

Эта команда запустит Docker, привяжет порт контейнера 3000 к тому же локальному порту, вызовет контейнер **lxf**, даст нам **tty**, сделает контейнер интерактивным (т.е. когда вы закончите работу с ним, он закроется), определит использование нашего тестового образа rails и, наконец, переведет нас в оболочку *bash* в самом контейнере.

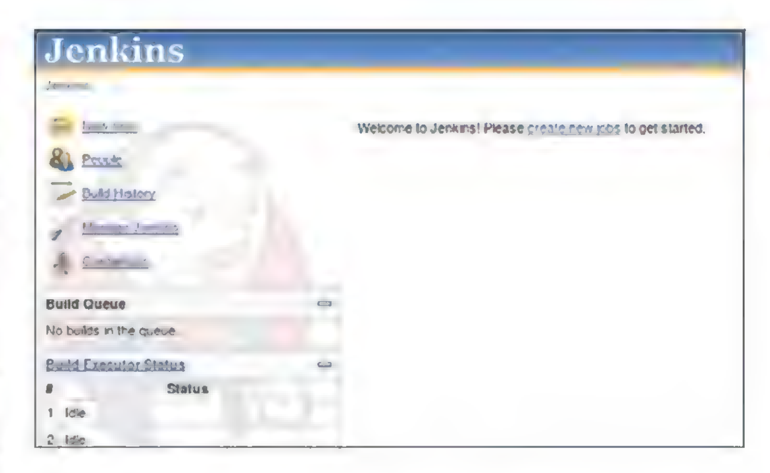
Отсюда мы можем запустить rails. В Dockerfile мы дали rails команду создать новое приложение в **/root/lxf**. Теперь, войдя через **cd** в эту директорию, мы можем дать команду серверу Rails:

```
cd /root/lxf
rails server
```

Rails по умолчанию настроен на использование *WEBrick*, небольшого легковесного сервера ruby HTTP Server, хорошо подходящего для сред разработки. Он запускается на порте 3000. Поскольку для этого порта мы дали команду соединения в нашем контейнере с нашим хостом, мы можем соединиться с ним через http://127.0.0.1:3000 с нашего рабочего стола. Появится знакомый экран Rails по умолчанию (на рис. внизу стр. 74).

Возможности Docker

Может, оно и не выглядит потрясающим, но сила Docker — в его возможности взять готовый контейнер и применять его где угодно. Пока что в плане настройки Rails мы сделали голый минимум, но можем кое-что добавить, создавая основу для наших команд разработки, пригодную для тестирования своего кода. К созданным таким способом образам есть несколько опций предоставления распределенного доступа. Docker Inc предлагает свой Hub — https://hub.docker.com (образы см. на http://registry.hub.docker.com), с возможностью свободного использования (и платными опциями для нескольких частных репозиториев). Однако иногда не разрешается выводить код за границы/сеть компании. В подобном случае образы можно хранить как обычные файлы, которые, в свою очередь, скопировать на целевой компьютер



> **Jenkins, инструмент CI с тоннами опций и плагинов (кое-кто даже может счесть, что их многовато).**

и без проблем запускать оттуда. Здесь мы сохранили копию тестового образа Rails в локальной директории:

```
sudo docker save -o ./railstest railstest
```

Полученный в результате файл (обычно архив TAR) можно взять и перенести, куда нам заблагорассудится. Желая запустить его на другом компьютере, где установлен Docker, мы просто используем команду **load**:

```
sudo docker load -i railstest
```

Отсюда мы можем в точности повторить вышеприведенные шаги, чтобы запустить сервер Rails на этой новой системе. Во время тестов мы перемещали этот образ между рабочим столом Ubuntu 14.04 и Centos 6.5 без малейших проблем. Однако есть еще одна опция: запуск реестра Docker локально в границах центра обработки данных или сети компании. Как и следует ожидать, у него открытый код, и он скачивается бесплатно — и как отдельное приложение, и, что проще всего, как образ Docker.

```
sudo docker pull registry
```

сделает для нас копию. В Registry [Реестр] масса опций: он может использовать хранение объектов — например, модуль Swift из OpenStack — чтобы держать в них образы Docker, но по умолчанию использует просто локальное хранилище. Для базового теста неплохо указать ему путь к хранилищу. Можно запустить его так:

```
sudo docker run -p 5000:5000 -v /tmp/registry:/tmp/registry registry
```

Это запустит наш локальный Registry и заставит его слушать порт 5000. Также тут велено Docker'у использовать локальную директорию как присоединенный том, чтобы данные в нем оставались после выключения контейнера. Теперь мы можем хранить в этом контейнере образы — начиная с нашей среды разработки railstest. Для сохранения образа послужит команда **docker push**. Однако по умолчанию она перемещает в глобальный репозиторий »

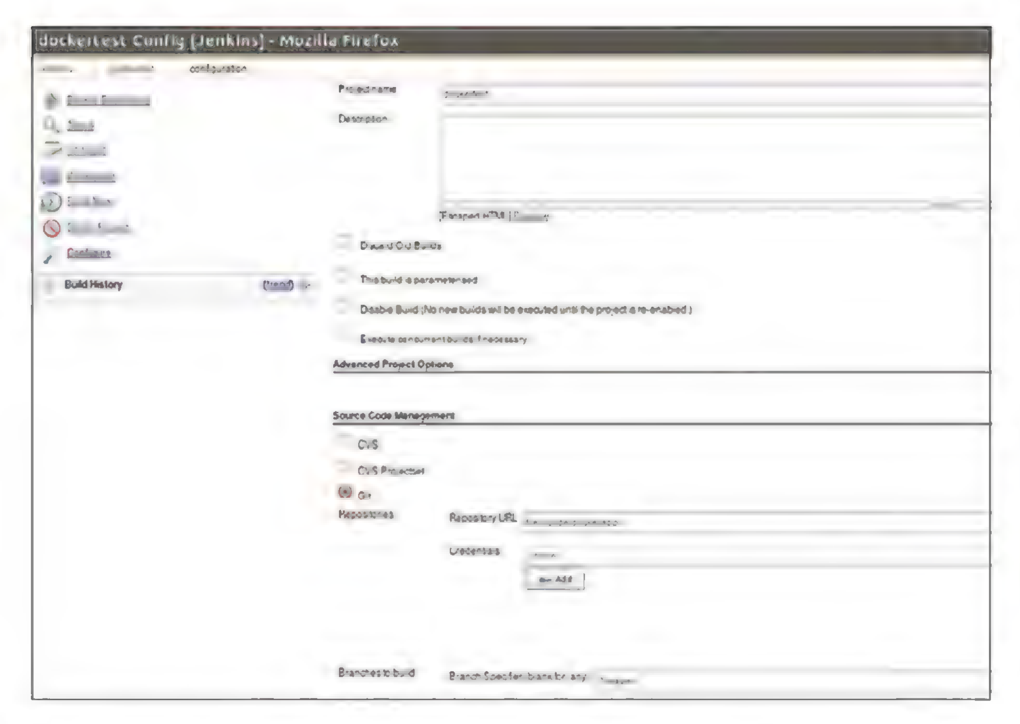
Скорая помощь

Если вам неудобно использовать *vi* в контейнере для редактирования файлов в командной строке, можете использовать другой редактор на вашем ПК и отредактировать их локально в **~/code/dockerapp**.

Непрерывный рост Docker

Docker и Docker, Inc (компания, образовавшаяся за проектом) весь прошлый год упоминалась в прессе, и это продолжилось в связи с объявлением о вложении в объеме $40 миллионов от Sequoia, известной VC-фирмы, которая поддерживала Google и множество иных известных имен в технологической индустрии. Говоря на языке стартапов (если, конечно, к Docker Inc этот термин еще приложим), перед компанией еще много взлетных полос, что позволит Docker улучшиться по сравнению с его текущим состоянием, когда это — формат для контейнеров со здоровой экосистемой участников, расширяющих его ежедневно (попробуйте, например, поискать Docker на GitHub). Docker все чаще внедряется другими платформами и проектами (Core OS, Apache Mesos), и его применяют весьма солидные компании (eBay, Spotify, Baidu). Основное внимание Docker Inc сосредоточено на превращение программы в более «готовую для производства» и, следовательно, более привлекательную для широкого применения — в качестве областей для улучшения упоминаются взаимодействие, создание кластеров, подготовка графиков работ, хранение и сетевое окружение. Когда эти красоты станут доступными, безусловно, проявится бурное неудовольствие по поводу расширения инструментов коммерческой поддержки и управления. Именно так Docker намерена зарабатывать деньги, и ее инвесторы надеются увидеть возврат своих вложений. Ставка делается на то, что Docker станет новым стандартом использования приложений «в облаке».

Однако код Docker остается открытым, и сообщество продолжит расширять и разрабатывать его всеми возможными способами, среди которых порой попадаются весьма неожиданные.



› Это первая часть работы нашего Jenkins, показывающая нам доступ к *Git* через fileshare.

Docker. Чтобы использовать локальный, нужно «маркировать» образ, указав его имя хоста/IP-адрес и порт. В новом терминале (поскольку в нашем исходном окне работает Docker Registry) надо ввести следующее:

```
sudo docker tag railstest localhost:5000/railstest
sudo docker push localist:5000/railstest
```

Это отправит шквал активности в оба окна — поток команд HTTP PUT в одном, где работает Registry, и состояние загрузки образа в другом. По завершении команды **sudo docker images** должно оказаться, что нам доступен наш новый образ **localhost:5000/railstest**. Мы также видим, что Registry использует свой том, отыскивая вновь созданные структуры файлов в **/tmp/registry**. Конечно, в реальной ситуации Registry находился бы на соответствующем сервере, доступном для всей нашей группы. Для данной задачи рекомендуется разместить его на web-сервере *Nginx* (или *Apache*). Ознакомьтесь с документацией по расширенным функциям на http://bit.ly/DockerRegistryAdvanced.

Теперь мы можем дать членам команды доступ к нашим Dockerfiles, и нужно разобраться с общим требованием к современным средам разработки, каковым является CI. Это касается (помимо многого другого) практики проведения автономных тестов в нашей базе кода, для гарантии, что когда новый код добавляется в систему, она не рухнет и не выдаст ошибок. CI (и ее близкий родственник, Continuous Delivery [непрерывная поставка]) — довольно обширная тема, глубокий анализ которой вне целей нашего урока. Однако на данный момент давайте предположим, что перед нами стоит задача запустить одну из общих систем CI с открытым кодом; и мы воспользуемся Jenkins. Он довольно широко используется, и за ним стоит большое сообщество. В проекте до Docker это могло бы означать установку нового сервера, установку JDK (Jenkins, написанного на Java) и затем скачивание программ Jenkins. Но коли у нас есть Docker, создание базовой системы Jenkins сводится просто к

```
sudo docker pull jenkins
sudo docker run —name localjenkins -p 8080:8080 -v /var/jenkins_home jenkins
```

После скачивания образа Jenkins из Docker Hub (на что требуется время), мы запустили сервер Jenkins на порте 8080 и добавили постоянный том для его данных. Заметьте, что мы не направили его на локальное хранение (хотя могли с легкостью это сделать, используя синтаксис из предыдущего примера), но можем скопировать данные из системы Jenkins (или любого другого контейнера Docker для этой цели) командой **docker cp**.

Хронический Docker

В некоторых средах Jenkins может проводить целые пакеты тестов при создании приложения. В конце этого процесса генерируются пакеты или исполняемые файлы; иногда применяются виртуальные машины, и этот код тестируется на них. Чудесно было бы раскрутить контейнер для этой цели, раз уж Docker потребляет так мало ресурсов! И вот бы затем использовать Jenkins для импорта успешной сборки в наш локальный Docker Registry... А почему бы и нет? Отключите на минутку контейнер Jenkins (просто нажмите Ctrl+C в его окне). Мы к нему еще вернемся. Прежде чем идти дальше, нужно разрешить Docker слушать удаленные команды наравне с сокетами, которые он слушает по умолчанию. В реальном сценарии понадобилось бы дополнительно позаботиться о безопасности, однако для урока сойдет и так. Через **sudo** отредактируйте файл **/etc/default/docker**, добавив следующую строку:

```
DOCKER_OPTS="-H 0.0.0.0:4243 -H unix:///var/run/docker.sock"
```

Давайте имитируем нашу первую проверку кода Ruby on Rails, написанного нашей командой с помощью нашего контейнера

Микросервисы

Термин «микросервисы» — это название программной архитектуры, не так давно обретшей большую популярность. Идея ее в замене монолитных приложений, которые обычно являются средним уровнем приложений предприятия «n-tier»; элементы, которые сидят между интерфейсом клиента (часто им является браузер) и базой данных движка с отдельными элементами, выполняющими отдельные более мелкие задачи.

В чем-то это повторяет идею Unix — множество мелких приложений, которые очень хорошо делают нечто одно. В монолитный приложениях, говорят сторонники микросервисов, внесение изменений становится со временем все труднее, поскольку даже небольшое изменение одного элемента может вызвать необходимость переделки всего приложения. Масштабирование также может стать дорогостоящим для крупных приложений и в какой-то мере избыточным, особенно в тех случаях, когда только один элемент требует масштабирования, но задействуется вся структура.

Подход микросервисов предполагает, что отдельные элементы являются отдельными сервисами, что позволяет использовать их независимо друг от друга и масштабировать по мере нужды. Эти сервисы общаются через границы между собой с помощью хорошо известных опубликованных интерфейсов.

Кроме того, разработка микросервисов может вестись небольшими группами, с теми инструментами и на тех языках, которые, по их мнению, приведут к оптимальным результатам для данного конкретного случая. Они, безусловно, более отказоустойчивы, и их намного проще заменять, избегая проблем «преемственности».

Противники этой концепции отвергают микросервисы, как «хипстерские SOA» (service oriented architecture — ориентированная на сервис архитектура), и указывают на уровень сложности, который они вносят в инфраструктуру.

Если отвлечься от споров об архитектуре, становится понятно, что Docker использовал ви́дение сторонников микросервисов и ожидает быстрого внедрения подобного типа проектов, что выглядит разумным, поскольку для работы с такими спецприложениями Docker — естественный выбор.

» **Пропустили номер?** Узнайте на с. 108, как получить его прямо сейчас.

Rails, который здесь является базовой структурой Rails, созданной нами в первом Dockerfile.

Сначала создадим локальную директорию для хранения кода — пусть мы в нашей директории **home**, и с этим справится простая команда **mkdir code**. Затем заново употребим наш образ railstest, который мы уже использовали ранее:

```
sudo docker run -p 3000:3000 --name railsdev -v ~/code:/
code -t -i railstest /bin/bash
```

Это вернет нас в командную строку в новом контейнере, с которым наша директория «кода» разделена как **/code**. Давайте скопируем наше исходное приложение Rails и запустим проверку источника изначального кода. В команды особо не вникайте — это же учебник не по Rails!

Проверка нашего приложения

Перед использованием Jenkins мы должны выполнить еще одну задачу — а именно, создать репозиторий *Git*, для совместного использования. В реальной жизни это, очевидно, был бы другой сервер (или контейнер Docker), или, возможно, интернет-сервис типа GitHub. Мы должны перейти в **/test/dockerapp directory**, и тогда просто надо дать следующие команды (по необходимости заменив указанные адрес электронной почты и имя):

```
cd /code
cp -r /root/dockerapp .
cd dockerapp
git init
touch git-daemon-export-ok
mkdir docker
git add .
git config ---global user.email "sysadmin@linuxformat.co.uk"
git config ---global user.name "Sidney Sysadmin"
git commit -m "initial check in"
```

Создастся новый репозиторий *Git* на нашем локальном диске, содержащий полную структуру нашего нового приложения Rails. В наши планы входит заставить Jenkins прочитать наш репозиторий, создать новый образ Docker из расположенного там Dockerfile и установить его, проверяя в процессе остальную часть нашего кода. Опять же, в реальной жизни Dockerfiles и код приложений располагались бы в разных репозиториях, однако ради экономии места на сей раз давайте создадим новый Dockerfile в поддиректории для Docker в нашем существующем репозитории:

```
FROM railstest
ENV HOME /root
RUN cd $HOME; rm -fr dockerapp
RUN git clone git://<ip address of our desktop>/dockerapp
```

Проверьте свое редактирование с помощью **git add .** и **git commit -m "added Dockerfile"**.

Мы можем использовать простой сервер *Git*, чтобы позволить новым контейнерам Docker считывать эти файлы. Откройте терминал, **cd** в директорию **~/code/dockerapp** и запустите

```
sudo git daemon ---reuseaddr ---base-path=/home/<ваше имя>/
code ---export-all
```

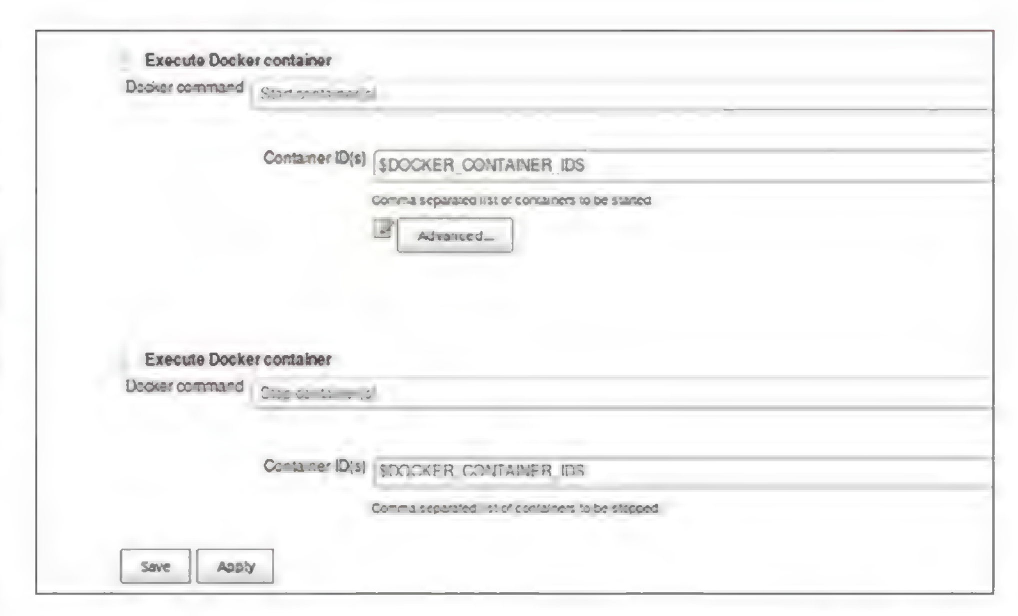
Пока что оставьте это окно открытым.

Jenkins, создай мне контейнер Docker

В отдельном окне снова запустите Jenkins, удалив ранее использованный контейнер, если мы хотим использовать для нового то же самое имя:

```
sudo docker rm localjenkins
sudo docker run —name localjenkins -p 8080:8080 -v /var/
jenkins_home -v ~/code:/code jenkins
```

Запустив браузер на нашем локальном рабочем столе, мы можем соединиться с http://127.0.0.1:8080, и перед нами откроется наша страница Jenkins. Сначала надо установить плагин для *Git*, чтобы Jenkins мог читать наш код. Щелкните по ссылке Manage



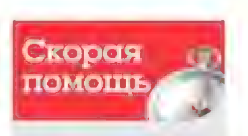
› **Вторая часть нашей работы с Jenkins, показывающая, как мы взаимодействуем с Docker и переменными среды.**

Jenkins в левой части экрана, затем по Manage Plugins, четвертой опции на появившемся экране. Щелкните по вкладке Available, затем по окну фильтра поиска (вверху справа), введите **Git Plugin**. Вы увидите много-много доступных для Jenkins плагинов. Выберите Git Plugin, отметив окошко рядом с его именем, и затем нажмите Install without restart [Установить без перезагрузки]. После завершения надо снова перейти в Plugins (на этот раз левая сторона), выбрать вкладку Available и найти docker. Нам нужен Docker Build. Повторим процедуру: выделите его и установите без перезагрузки. И — по новой, для другого плагина, TokenMacro.

После этого вернемся к ссылке Manage Jenkins и выберем систему настройки на следующем экране. Нам нужно настроить наш плагин Docker и убедиться, что мы в состоянии общаться с нашим хостом Docker (в данном случае это наш настольный ПК). Прокрутите экран вниз и введите в Docker URL — **http://<IP-адрес вашего настольного ПК>:4243**. Здесь мы можем проверить соединение с помощью кнопки с соответствующим названием, и должны получить радостную весть Connected to... [Соединение с...]. Если все нормально, нажмите Save [Сохранить].

Теперь можно нажать на ссылку Create new jobs [Создать новое задание], перед нажатием OK поименовав наше задание dockertest и выбрав опцию Свободный программный проект [free-style software project]. Для опции Source Code Management [Управление исходником] можно благодаря нашему плагину выбрать *Git*, и ввести URL **file:///code/dockerapp** (логин и пароль не нужны).

Сместившись по экрану вниз, можно добавить шаг сборки, выбрав Execute Docker Container [Выполнить контейнер] из выпадающего списка (рис. вверху на стр. 76). Здесь мы выберем Create Image [Создать образ]. Опции по умолчанию для контекстной папки вполне нормальные, поскольку нам просто нужно изменить тэг, добавив к нему спереди rails_. Добавьте второй шаг по созданию, на сей раз по созданию контейнера; Image name — это тэг предыдущего шага. Здесь имя хоста может быть любым. Добавьте еще один шаг, на этот раз запуская контейнер Docker с ID $DOCKER_CONTAINER_IDS (это переменная среды из плагина). Наконец, добавьте шаг, где действием будет остановка контейнеров. И снова, $DOCKER_CONTAINER_IDS является здесь значением поля. Когда все будет включено, сохраните процедуру и выберите опцию Build Now [Собрать] в левой стороне. Jenkins выгрузит наш файл Docker, создаст образ, запустит образ, и, по подтверждении его успешности, закроет его. Посмотрите на статус процедуры — если он красный, это плохо, если синий — хорошо! — и на то, какие выполняемые действия отображаются в консоли. **sudo docker images** покажет, что для использования доступен образ rails_*. Эту простую процедуру можно принять за основу системы CI, использующей Docker, и расширить ее, включив большее количество кода и тестирование приложений. Наслаждайтесь! LXF



Вы можете запустить задание сборки Jenkins в любой момент — просто сохраните и нажмите Build Now. Экспериментируйте с разными опциями, да следите, какие появятся сообщения об ошибках!

Метаданные: Исследование файлов

Прежде чем бурно опротестовывать саму идею, гляньте, что предлагает **Шон Конвэй** по метаданным файлов и двум приложениям для их исследования.



Наш эксперт

Шон Конвэй обеспечивает безопасность данных для систем телекоммуникации, и работает в области поддержания жизненного цикла компьютерных систем. Поскольку он лично столкнулся с похищением персональных данных, сохранение безопасности таковых стало для него персональным мотивом.

Сохраняя файл, вы сохраняете пользовательские данные, то есть данные, которые вас интересуют. Обычные примеры подобных пользовательских данных — тексты, чертежи, изображения и музыкальные листинги, но вместе с этими видами данных хранятся еще и метаданные, являющие собой данные о данных. Метаданные — это информация, хранящаяся в файле и предоставляющая сведения о свойствах самого файла. Например, один из типов метаданных файла — права собственности на документ.

В порядке аналогии, представьте себе книгу, которая содержит пользовательские данные и метаданные. Уведомление об авторских правах, содержание и глоссарий — это некоторая часть метаданных (данных о данных) по содержанию (пользовательских данных) книги. Метаданные файла могут содержать личную информацию, скрытые сведения, свойства сервисов, координаты местоположения, сведения об изменениях документа, другие данные, поддерживаемые приложением, и подробности об устройстве (например, фотокамере), создавшем файл. По умолчанию Flickr и Picasa Web Albums удаляют информацию геолокации из подгруженных фото по соображениям безопасности, и о включении этой информации вы должны указать явно.

Одним из приложений, рассматривающих метеданные файлов, является *Exiftool*. Процитируем Фила Харви [Phil Harvey], разработчика: «*ExifTool* — независимая от платформ библиотека Perl и приложение командной строки для чтения, записи и редактирования мета-информации самых разнообразных файлов». Установка его очень проста: для Fedora 19 скомандуйте

```
sudo yum install perl-Image-ExifTool
```

```
sconway@fedora19-inane-ca:~/Documents/writing_workstation/computer writings/metadata
File  Edit  View  Search  Terminal  Help

    File Types
    3FR   r      | EIP    r      | LA     r      | OTF    r      | RW2     r/w
    3G2   r/w    | EPS    r/w    | LNK    r      | PAC    r      | RWL     r/w
    3GP   r/w    | ERF    r/w    | M2TS   r      | PAGES  r      | RWZ     r
    ACR   r      | EXE    r      | M4A/V  r/w    | PBM    r/w    | RM      r
    AFM   r      | EXIF   r/w/c  | MEF    r/w    | PCD    r      | SEQ     r
    AI    r/w    | EXR    r      | MIE    r/w/c  | PDF    r/w    | SO      r
    AIFF  r      | EXV    r/w/c  | MIFF   r      | PEF    r/w    | SR2     r/w
    APE   r      | F4A/V  r/w    | MKA    r      | PFA    r      | SRF     r
    ARW   r/w    | FFF    r/w    | MKS    r      | PFB    r      | SRW     r/w
    ASF   r      | FLA    r      | MKV    r      | PFM    r      | SVG     r
    AVI   r      | FLAC   r      | MNG    r/w    | PGF    r      | SWF     r
    BMP   r      | FLV    r      | MODD   r      | PGM    r/w    | THM     r/w
    BTF   r      | FPF    r      | MOS    r/w    | PLIST  r      | TIFF    r/w
    CHM   r      | FPX    r      | MOV    r/w    | PICT   r      | TORRENT r
    COS   r      | GIF    r/w    | MP3    r      | PMP    r      | TTC     r
    CR2   r/w    | GZ     r      | MP4    r/w    | PNG    r/w    | TTF     r
    CRW   r/w    | HDP    r/w    | MPC    r      | PPM    r/w    | VRD     r/w/c
    CS1   r/w    | HDR    r      | MPG    r      | PPT    r      | VSD     r
```

› **Введя в терминале man up exiftool, вы увидите поддерживаемые функции и типы файлов.**

В разработке

Наша работа в *Linux Format* — предоставлять вам, наши замечательные читатели, руководства по работе с избранными нами программами с открытым кодом. В данном случае приложения находятся на стадии активной разработки, и есть немалый риск, что они не будут работать так, как описано. Прочитав документацию по *Exiftool* и *MAT*, вы увидите там ясное предупреждение об этом. Не то что разработчики обоих инструментов считают свои инструменты ненадежными; дело в том, что метаданные — область необычная.

а для Ubuntu-разновидности Linux:

```
sudo apt-get install libimage-exiftool-perl
```

Если вы относитесь к пуристам и считаете, что единственно достойный путь — это сборка с нуля, то на официальном сайте (http://bit.ly/ExifToolDL) вы найдете исходный код и инструкции по созданию собственной сборки.

Exiftool — это приложение командной строки (CLI), и его man-страница (**man exiftool**) предоставляет обширную документацию. В ней содержатся таблицы поддерживаемых на текущий момент типов файлов и три разные функции: read, write и create, которые доступны не для всех типов файлов. Однако Харви говорит, что поскольку потребности определены, над ними ведется работа, и большая часть работы проекта касается метаданных, которые камеры добавляют в файлы фотографий.

Чтобы исследовать метаданные по файлу, наберите

```
exiftool file_type.png
```

Очистит метаданные файла команда

```
exiftool -all= file_type.png
```

При записи информации приложение сохраняет исходный файл, добавляя к имени файла суффикс **_original**. В примере выше *Exiftool* удаляет все метаданные в тэгах, связанных с файлом. Загляните на официальный сайт, там есть дополнительные примеры по настройке процесса удаления.

Одна из сложностей с метаданными файлов в том, что они бывают не документированы. Метаданные не хранятся в определенном месте файла, например, в его конце; они могут быть разбросаны по всему файлу. Имена метаполей тоже не являются постоянными, и различаются в зависимости от поставщика.

Некоторые из метаданных файлов являются обязательными, и их нельзя менять или удалять, поскольку они предоставляют необходимую для поддержки файла информацию. Поэтому удаление всех метаданных сделает файл нечитаемым.

Большая часть исследований по определению метаданных файлов делается средствами обратного инжиниринга. Это очень напряженный процесс, требующий немало времени и тестирования. Сложность метаданных файлов — одна из причин того, что только ограниченное число утилит их поддерживают.

Очистка метаданных с помощью *MAT*

Еще один инструмент для изучения метаданных — *Metadata Anonymisation Toolkit* или *MAT*. Этот инструмент поддерживается как через терминал, так и через графический пользовательский интерфейс.

Жюльен Вуазен [Julien Voisin] разработал *MAT* во время Summer of Code [Лето за кодом] при Google (GSoC) в 2011 г. GSoC — это ежегодная программа Google, созданная для продвижения разработки программ с открытым кодом во время летних отпусков и каникул. Это хорошо организованный процесс, который позволяет студентам заработать за свои труды деньги. Отчет об этой работе и более подробную информацию о *MAT* можно найти на сайте Arxiv Корнельского университета в виде PDF (http://bit.ly/MATHistory).

На момент написания этой статьи репозитория Fedora с установкой *MAT* не существует. Компиляция требует наличия исходного кода от разработчика и ряда зависимостей с других сайтов. Внятных инструкций по полной установке в Fedora нет.

На **LXFDVD** имеется файл для компиляции *MAT 0.5.2* в системах Linux на базе Fedora 19. Начните с установки RPM-пакета Python — *python-distutils-extra*. Затем установите пакеты *hachoir*, в следующем порядке: **hachoir-core-1.3.3.tar.gz**, **hachoir-subfile-0.5.3.tar.gz**, **hachoir-metadata-1.3.3.tar.gz**, **hachoir-urwid-1.1.tar.gz**, **hachoir-parser-1.3.4.tar.gz**, **hachoir-wx-0.3.tar.gz** и **hachoir-regex-1.0.5.tar.gz**. Используйте стабильный релиз *MAT* — **mat-0.5.2.tar.xz** с сайта поддержки (http://mat.boum.org). Распакуйте tar-архив и, перейдя в директорию **0.5.2**, наберите команду

```
python setup.py install
```

Она установит GUI *MAT* и предоставит инструмент CLI.

Установка *MAT 0.4.2* на ПК с Ubuntu Linux намного проще. Обратите внимание: цифры номера версии — не опечатка. Команда разработки *MAT* создает продукт, который нужно портировать в выбранные сообщества. *MAT 0.4.2* — это версия, используемая для установки сообществом Ubuntu:

```
sudo apt-get mat
```

Для просмотра метаданных файла в терминале наберите

```
mat -d file_type.png
```

Для очистки метаданных файла с помощью CLI скомандуйте

```
mat -f file_type.png
```

Команда **mat** с атрибутом **-d** покажет метаполя файла, которые разработчики инструмента отметили как вредные. Если вас интересуют поддерживаемые типы файлов, эту информацию предоставит инструмент, вызываемый с помощью атрибута **-l**.

Во время одного из наших тестов инструмент *MAT* выдал ошибку, и после общения с разработчиком эта ошибка стала считаться признанной. Дальнейшие тестовые сценарии с инструментом *MAT* делали некоторые файлы документов нечитаемыми для родительского приложения, после того, как **MAT** производил очистку без ошибок. Если вы хотите заниматься очисткой файлов документов, внимательно изучите, какие приложения поддерживаются. На данной стадии разработки наблюдается некое нежелание заниматься модификацией метаданных документа для этого типа приложений. Если вы все же решились на риск, то, пожалуйста, сделайте резервную копию. Нам даже подумать страшно о том, что очистка метаданных может привести к такой проблеме, как потеря чего-то важного, например, дипломной работы.

LibreOffice, свободный офисный пакет, предлагает опции по поддержке безопасности документа, которые, в отличие от остальных упомянутых нами инструментов, могут удалять метаданные из файлов программы. Вы можете включить эту функцию из панели инструментов меню, выбрав ее в Tools > Options > Security > Options. Еще один инструмент, предлагающий опцию удаления метаданных — *Shotwell* (0.14.1). Этот инструмент управления фотографиями предлагает радиокнопку, удаляющую метаданные при экспорте фотографий.

Если вы хотите удалять метаданные в других приложениях, загляните в документацию по каждому из них. К сожалению, вы обнаружите, что некоторые приложения этого даже не рассматривают. Метаданные считаются полезными как для пользователя, так и для приложения. Например, метаданные с информацией о датах и авторе изменений очень удобны для отслеживания документов.

Не обязательно считать, что метаданные несут в себе угрозу, однако пользователи, распространяющие инофрмацию, должны в полной мере осознавать, что именно содержат в себе их файлы и есть ли риск в раскрытии этих данных. Оба упомянутых инструмента позволяют пользователям принять такое решение, да еще и отредактировать метаданные, так что принимайтесь за исследования! LXF

Не все MAT'ы равны

Во время написания данного руководства обнаружились некоторые противоречия между вариантами инструмента *MAT* для Fedora и Ubuntu. Версия Fedora не очищает файлы *LibreOffice* или *Microsoft Office 2010*, а инструмент Ubuntu справляется с этим.

Изучение соответствующей man-страницы показало, что ряд атрибутов, поддерживаемых в версии 0.4.2, не поддерживается в версии 0.5.4. Учитывая, что уровень версии Fedora данного инструмента выше, это выглядит шагом назад.





Скорая помощь

Обратите внимание, что инструменты, о которых мы рассказали, написаны на Perl или Python, и чтобы приложения установились нормально, соблюдите все условия для каждого языка.

	A	B	C
1		Before exiftool	After Exiftool
2	ExifTool Version Number	9.6	9.6
3	File Name	file_types.png	file_types.png
4	Directory	.	.
5	File Size	115 kB	115 kB
6	File Modification Date/Time	2014 9 01 11 30 07-05 0	2014 9 01 11 30 56-05
7	File Access Date/Time	2014 9 01 11 30 07-05 0	2014 9 01 11 30 56-05
8	File Inode Change Date/Time	2014 9 01 11 30 07-05 0	2014 9 01 11 30 56-05
9	File Permissions	rw-rw-r--	rw-rw-r--
10	File Type	PNG	PNG
11	MIME Type	image/png	image/png
12	Image Width	897	897

› К счастью, в данном экранном снимке не применялось никакого незаконного ПО.

Контролируйте свою систему

Михалис Цукалос объясняет главное: что нужно знать для мониторинга вашей системы Linux и диагностики проблем производительности.



Наш эксперт

Михалис Цукалос получает истинное удовольствие от администрирования UNIX и программирования на C, Perl, Erlang и Swift. Кроме того, он DBA и математик.



Мониторинг компьютерной системы означает изучение всех аспектов системы, способных повлиять на ее производительность, стабильность и ровную работу. По большей части мониторинг проводится для решения проблем производительности. Процесс мониторинга охватывает мониторинг программ и оборудования.

Визуализация — это удобный способ просмотра данных мониторинга с целью обнаружения аномалий, ведущий к определению и решению проблем. Решение проблем с производительностью, обнаруженных в процессе мониторинга — не всегда простая задача, и она может потребовать дополнительного мониторинга и экспериментов. Как и при отладке программ, самые сложные проблемы — те, которые не происходят регулярно или не воспроизводятся.

Мониторинг данных может быть в реальном времени или по предыстории. «Исторический» не всегда подразумевает данные двухмесячного возраста: иногда это данные, которым всего 10 минут (тогда он довольно близок к мониторингу в реальном времени). Правильный инструмент для определения проблем производительности выбрать нелегко, поскольку разные инструменты дают разную информацию. Администраторы должны быть знакомы со многими инструментами и обладать умением выбрать подходящий, на основе собственного опыте и рекомендаций других администраторов.

Решение проблем производительности предполагает разные шаги. Во-первых, вы должны понимать, что проблема у вас есть. Далее, вы должны быть в состоянии ее воспроизвести. Третий шаг подразумевает выбор правильного инструмента мониторинга для сбора полезной и уместной информации. Затем вам нужно интерпретировать данные мониторинга и определить проблему. Последние шаги — решить проблему, и, что крайне важно, убедиться, что проблема действительно решена.

Наш урок представит инструменты и технологии мониторинга посредством традиционных инструментов Unix, которые найдутся в любой установке Linux, и мы проведем мониторинг элементов, имеющих отношение к общей стабильности и здоровью системы Linux. Дальнейшие уроки продемонстрируют более современные и специализированные инструменты и технологии.

Мониторинг чего и как?

Решение о том, что подвергать мониторингу — не самая простая задача, однако вы всегда можете изменить свой подход к мониторингу. Мой личный подход предполагает первым делом, прежде чем вдаваться в подробности, проанализировать общую производительность системы Linux. Я также считаю, что процесс мониторинга должен начинаться после настройки системы Linux. Поэтому неплохой стратегией будет для начала выполнить мониторинг загрузки системы, памяти, объема подкачки и сетевых соединений, а уж потом переходить к чему-то еще. Визуализация данных быстро даст вам информативное представление собранных данных, что поможет вам в выявлении проблем или неполадок.

Второй важнейший вопрос — как проводить мониторинг. Обычно самый разумный подход — мониторинг с помощью инструмента, который вам лучше всего знаком. Альтернатива — взять самый простой инструмент из способных справиться с этой задачей.

Хранят данные мониторинга двумя основными способами: в виде простого текста или с помощью базы данных. Первый способ проще, но по мере увеличения размера текстового файла усложняется и управление им. Эту проблему можно решить, используя несколько текстовых файлов, однако тогда придется читать и осуществлять поиск по многим текстовым файлам, чтобы просмотреть свои данные. Преимущества использования текстовых файлов в том, что их легко обрабатывать и изменять с помощью традиционных инструментов Unix (*grep*, *wc*, *awk*, *sed* и т.п.), а значит, много новых инструментов изучать не придется. Второй способ, использование базы данных, труднее в реализации, и, в зависимости от количества данных, может потребовать наличия еще одного человека для администрирования базы

Скорая помощь

Помните, что определить проблему — еще не значит ее решить. Сам по себе мониторинг ни одной проблемы не решает. Одно дело — выяснить, что ваш сайт работает медленно, и совершенно другое — понять, что причина торможения — нехватка ОЗУ на вашем сервере базы данных.

Sar и sysstat

Еще один отличный ресурс для получения информации мониторинга — инструмент мониторинга производительности *sar*. *Sar* дает вам ту же самую информацию, что и любой другой инструмент или технология, но его преимущество в том, что у него есть собственное управление файлами и собственные инструменты отчетности, и он может сообщать хронологические данные. *Sar* входит в пакет *Sysstat* Unix — это не отдельный инструмент, а обыкновенная утилита, взаимодействующая с пользователями.

Чтобы запустить его на системе Debian 7, сначала надо изменить показатель ENABLED на **true** в файле **/etc/default/sysstat**, чтобы программа *sadc* могла приняться за сбор данных по активности системы. Затем вам нужно запустить сервис sysstat. В системе Debian 7 для этого послужит следующая команда:

```
# /etc/init.d/sysstat start
[ ok ] Starting the system activity data collector: sadc.
```

Файлы с данными записываются в директорию **/var/log/sysstat/**. Для проверки хронологических данных использования CPU (пользуясь файлом **/var/log/sysstat/sa01**) запустите *sar* так:

```
$ sar -f /var/log/sysstat/sa01
```

Итак, *sysstat* и *sar* — хорошие альтернативы представленным технологиям сбора данных о производительности, но надо анализировать результат и применить R или подобный ему пакет для визуализации. Предложенные технологии просто предлагают бóльшую степень контроля за процессом сбора данных и позволяют легче вносить изменения. Выбор же методики — это дело личного вкуса.

данных. Хорошая сторона в том, что у вас появляется возможность делать запрос по данным с помощью SQL (предполагается, что вы уже знакомы с SQL), не заморачиваясь управлением хранением. Кроме того, доступ к базе данных намного проще получить с удаленной машины (см. учебник по RSysLog, **LXF181**), и обеспечивается бóльшая безопасность данных. Если база данных находится не на той машине, где проводится мониторинг, то ваши данные мониторинга уцелеют, даже если на «подопытной» машине произойдет сбой.

Инструменты, представляющие информацию в реальном времени (*top*, *tcpdump*, *ntop* и т.д.), наверное, хороши, но я полагаю, что лучше всего те технологии, которые позволяют вернуться назад и просмотреть ранее собранные данные, потому что это дает лучшее общее представление о работе системы. Кроме того, я полагаю, что разумнее сначала использовать стандартные инструменты Unix, а уж потом переходить к использованию чего-то более современного.

Представленные здесь технологии следует применять в качестве руководящих указаний, приспосабливая их к своим конкретным нуждам.

Мониторинг средней нагрузки

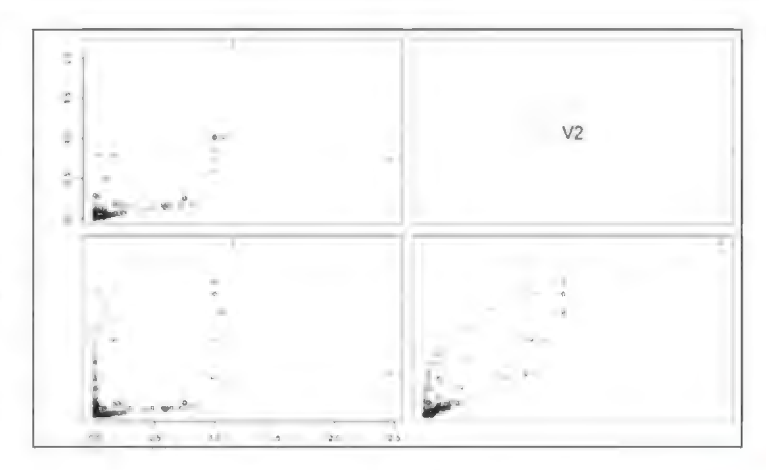
Простейший способ мониторинга средней нагрузки — использовать результат команды **uptime** и небольшой скрипт *awk*, который будет работать как задание *Cron*, сохраняя ваши данные в текстовом файле. Вот этот скрипт *awk*:

```
$ ls -l uptime.sh
-rwxr-xr-x 1 mtsouk mtsouk 85 Oct 4 21:07 uptime.sh
$ cat uptime.sh
#!/bin/bash
uptime | awk '{print $10 $11 $12}' | awk -F, '{print $1 " " $2 " " $3}'
$ ./uptime.sh
0.00 0.01 0.05
$ crontab -l
*/5 * * * * /home/mtsouk/bin/uptime.sh >> ~/uptime.data
```

Текстовый файл с данными должен выглядеть примерно так:

```
$ head -5 uptime.data
0.00 0.01 0.05
0.98 0.58 0.27
1.00 0.85 0.48
1.06 1.01 0.65
1.00 1.01 0.75
```

Первое число показывает среднюю нагрузку за последнюю минуту, второе — за последние 5 минут, и третье — за последние 15 минут. Эти три цифры покажут вам: увеличилась нагрузка, уменьшилась или осталась без изменений. Если нагрузка выше



› Рис. 1. Визуализация файла uptime.data с помощью R. R умеет генерировать весьма впечатляющие графики.

общего числа CPU или ядер, то ваша система страдает, и следует что-то предпринять. В использованной нами системе Linux только один CPU, поэтому любая средняя нагрузка выше 1.00 означает проблемы с производительностью (если появляется надолго). Показатель 0.60 означает, что за взятый период времени CPU работал 60 % всего времени, а оставшиеся 40 — не использовался, что весьма неплохо. Показатель 2.5 означает, что в среднем работает 2,5 процесса, и каждый должен быть запланирован в CPU. Поэтому CPU довольно сильно занят.

Показатели *uptime* весьма полезны для выяснения наличия в вашей системе проблем с производительностью, но нужно использовать и другие инструменты для обнаружения, понимания и решения реальных проблем. Полезные инструменты для обнаружения проблем производительности — *top*, *htop*, *lsof*, *netstat* и т.д.

Текстовый файл с данными *uptime* был обработан с помощью R следующим образом:

```
> data <- read.table(«~/uptime.data», header=FALSE)
> summary(data)
V1               V2               V3
Min. :0.00000    Min. :0.01000    Min. :0.05000
1st Qu.:0.00000  1st Qu.:0.01000  1st Qu.:0.05000
Median :0.00000  Median :0.01000  Median :0.05000
Mean :0.01688    Mean :0.02423    Mean :0.05681
3rd Qu.:0.00000  3rd Qu.:0.02000  3rd Qu.:0.05000
Max. :2.47000    Max. :2.15000    Max. :1.05000
> pairs(data)
```

Как вы понимаете, V1 — это первый столбец, V2 — второй, а V3 — третий. Команда **summary()** — отличный способ просмотра данных. Ее я запускаю первой во всех случаях. Команда **pairs(<dataset_name>)** организует все пары столбцов в наборы данных (см. рис. 1). Если у вас четыре столбца, то в выводе будет 16 ячеек.

Скорая помощь

» Никогда не пытайтесь решать проблемы, если вы устали или не выспались.
» Иногда лучше оставить работу на завтра и просто пойти прогуляться.
» Постарайтесь изолировать проблему.
» Попробуйте воспроизвести проблему.
» Прежде чем затевать мониторинг чего-то нового, мысленно сформулируйте ясную цель.

Если вы слабо знакомы со статистикой, не пропустите следующий номер *Linux Format*, где вы сможете больше узнать о результатах команды **summary()** и о том, как автоматизировать большую часть представленных задач.

Мониторинг дискового пространства

Есть множество способов провести мониторинг места на диске. Простейший — посредством утилиты командной строки *df*. В системах на Debian 7 с одним разделом на диске, вывод утилиты *df* при отображении файловых систем, отформатированных в ext3, такой:

```
$ df -t ext3
Filesystem 1K-blocks Used Available Use% Mounted on
/dev/root 24189340 9903888 14039216 42% /
```

Этот пример можно принять за основу для мониторинга дискового пространства. Если у вас больше разделов, добавьте их в вывод. Если использовать опцию **-m**, место на диске напечатается в мегабайтах, а не в килобайтах.

Полный код скрипта *diskUsage.sh* следующий:

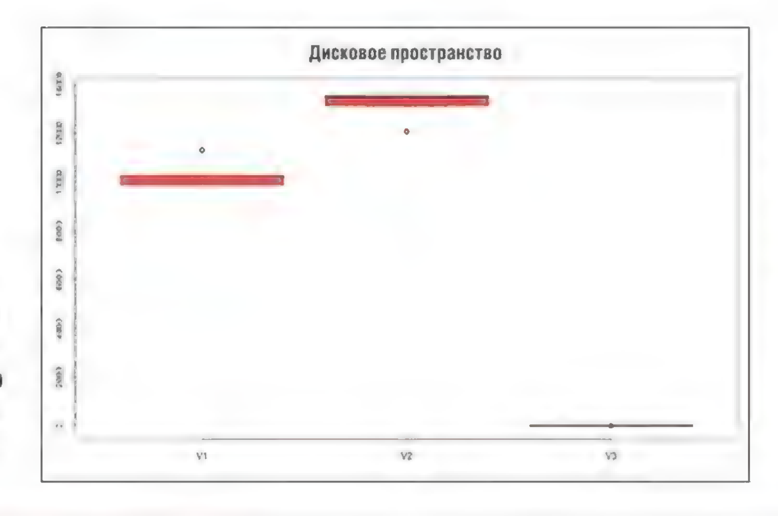
```
$ cat diskUsage.sh
#!/bin/bash
/bin/df -t ext3 -m | tail -1 | awk {,print $3 „ „ $4 „ „ $5'}
$ ./diskUsage.sh
9709 13673 42%
```

Скрипт будет запускаться как задание *Cron*, как ранее скрипт *uptime.sh*, и сохранять свои данные в текстовом файле **diskSpace.data**. На сей раз интерпретация результатов будет немного другой, потому что вам теперь важно не перейти заданный порог. Диск, заполненный на 90%, требует принятия немедленных мер, особенно если компьютер, где проводится мониторинг, является почтовым или FTP-сервером.

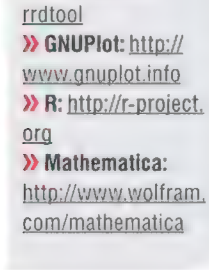
Текстовый файл **diskSpace.data** обрабатывается с помощью R следующим образом:

```
> data <- read.table(«~/diskSpace.data», header=FALSE)
> summary(data)
V1              V2              V3
Min. : 9709     Min. :12069     42%:288
1st Qu.: 9878   1st Qu.:13108   43%:405
Median : 9891   Median :13492   44%:453
Mean :10011     Mean :13372     45%: 44
3rd Qu.:10275   3rd Qu.:13505   49%: 6
Max. :11314     Max. :13673
> boxplot(data, main="Дисковое пространство", col="red")
> grid()
```

Для визуализации данных выбрана полосковая диаграмма (рис. 2). Она может предоставить информацию о форме, вариативности и медиане наборов статистических данных и позволяет быстро и четко выделить выбросы.



› Рис. 2. Визуализация файла diskSpace.data с помощью полосковой диаграммы в R. Вывод показывает, что значения довольно близкие, без особого разброса.



Среди полезных инструментов мониторинга и визуализации:
- » **MRTG:** http://oss.oetiker.ch/mrtg
- » **Cacti:** http://www.cacti.net
- » **RRDTool:** http://oss.oetiker.ch/rrdtool
- » **GNUPlot:** http://www.gnuplot.info
- » **R:** http://r-project.org
- » **Mathematica:** http://www.wolfram.com/mathematica

› Рис. 3. Диаграмма, показывающая общее число установленных TCP-соединений по часам суток.

Мониторинг активных TCP-соединений

Есть два простых способа провести мониторинг ваших активных TCP-соединений. Первый — с помощью *netstat*, а второй — с помощью *lsof*. Проблема с *lsof* в том, что тут нужны привилегии root, поэтому *netstat* предпочтительнее, если только вы не хотите запустить задание *Cron* от имени root.

```
$ netstat -nt | tail -n +3 | wc -l
3
# lsof -nP -iTCP -sTCP:ESTABLISHED | tail -n +2 | wc –l
3
```

Знание активных соединений на сервере позволит оценить общее состояние всей системы. Малое число активных соединений означает, что ваша система быстро обрабатывает запросы. Большое число активных соединений в сочетании с большой средней нагрузкой означает, что ваша система работает медленно из-за изобилия соединений TCP, обслуживаемых недостаточно быстро. Аналогично, рост числа установленных соединений (даже без особой нагрузки на систему) — это явление, которое стоит исследовать подробнее.

Следующий скрипт (*tcpConnect.sh*) использует *netstat* для записи числа соединений, а также даты и времени измерения:

```
$ cat tcpConnect.sh
#!/bin/bash
C=$(/bin/netstat -nt | tail -n +3 | grep ESTABLISHED | wc -l)
D=$(date +"%m %d")
T=$(date +"%H %M")
printf "%s %s %s\n" "$C" "$D" "$T"
$ ./tcpConnect.sh
3 10 06 22 22
$ crontab -l
*/5 * * * * /home/mtsouk/bin/tcpConnect.sh >> ~/connections.
data
```

Текстовый файл **connections.data** был обработан с помощью R следующим образом:

```
> data <- read.table("~/connections.data", header=FALSE)
> summary(data$V1)
Min.     1st Qu.   Median   Mean     3rd Qu.   Max.
0.0000   0.0000    1.0000   0.8847   1.0000    26.0000
> newData <- tapply(data$V1, list(data$V4), sum)
> newData
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
102 55 38 1 35 23 20 21 89 170 175 153 77 109 100 84 65 57 62
127 165 140 124 257
> barplot(newData, xlab="Час", ylab="Количество", col="yellow",
las=2, border="red", main="Соединений за данный час суток")
> grid()
```

Как видно по результату, максимальное количество соединений — 26, но по большей части машина не столь занята (средний показатель невысок). Команда **barplot** выдает удобный график (рис. 3), показывающий общее число соединений в час суток с помощью новой переменной с именем **newData**. Эта переменная хитроумно объединяет данные из двух столбцов (**data$V1** и **data$V4**), чтобы выдать искомый результат. С помощью той же технологии вы можете узнать общее число соединений за месяц или за определенный день месяца. Если вы видите большое число соединений в необычное время, это может означать попытку взлома.

Мониторинг ОЗУ и области подкачки

Простейший способ провести мониторинг ОЗУ и области подкачки — применить команду **free**:

```
$ free
        total    used    free    shared  buffers  cached
Mem:  1015088  834680  180408   0        120352   315716
-/+ buffers/cache: 398612 616476
Swap: 524284 23612 500672
```

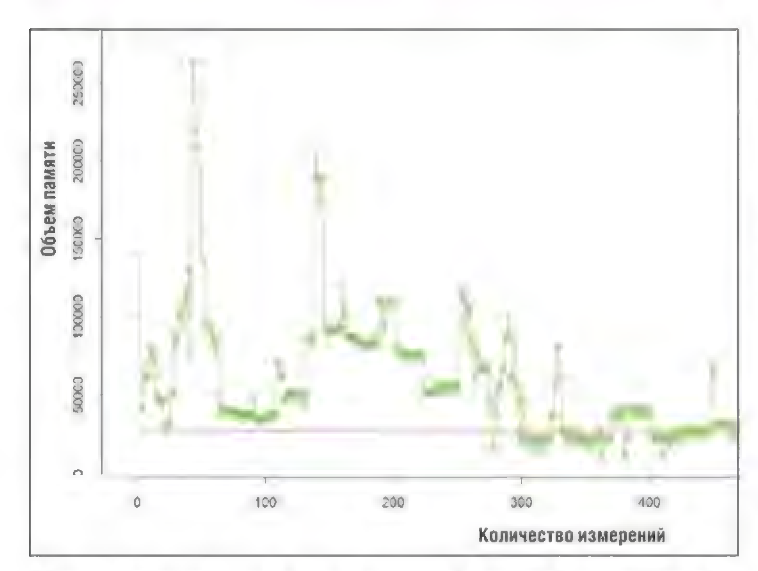
Для целей нашего урока мы проанализируем два показателя: объем свободной памяти и используемую область подкачки [swap]. Низкое количество свободной памяти — индикатор того, что систем приходится выполнять много задач; это может привести к подкачке диска, что отнюдь не здорово. Активное использование области подкачки заставляет систему тормозить, потому что доступ к диску во много раз медленнее, чем к памяти.

Следующий скрипт (*memory.sh*) записывает два показателя в файл **memory.data file**:

```
#!/bin/bash
F=$(/usr/bin/free | head -2 | tail -n +2 | awk {'print $4'})
S=$(/usr/bin/free | head -4 | tail -n +4 | awk {'print $3'})
printf "%s %s\n" "$F" "$S"
$ ./memory.sh
180408 23612
$ crontab -l
*/2 * * * * /home/mtsouk/bin/memory.sh >> ~/memory.data
```

Текстовый файл **memory.data** был обработан с помощью R следующим образом:

```
> data <- read.table("~/memory.data", header = FALSE)
> id <- rownames(data)
> head(id)
[1] "1" "2" "3" "4" "5" "6"
> plot(x=id, y=data$V1, xlab="Количество измерений",
ylab="Объем памяти", col="green")
> lines(x=id, y=data$V1, xlab="Количество измерений",
ylab="Объем памяти", col="green")
> lines(x=id, y=data$V2, col="red")
```



› Рис. 4. Визуализация объема свободной памяти и используемой области подкачки с помощью R.

Утилита netstat

Netstat — стандартная утилита Unix, которая быстро показывает исчерпывающую информацию о статусе сети. Это очень мощный инструмент, работающий на уровне Socket, TCP, UDP, IP и Ethernet. Его недостаток в том, что он показывает только информацию о локальной машине; утилиты вроде *nmap* и *tcpdump* умеют собирать информацию о машинах, находящихся в той же локальной сети. Следующая команда покажет множество статистических данных по каждому сетевому протоколу, но для поиска интересующей вас информации следует применить *grep*:

```
$ netstat -s
```

Следующая команда *netstat* показывает информацию о соединениях конкретного протокола (ssh):

```
$ netstat -a | grep -i ssh
tcp 0 0 *:ssh *:* LISTEN
tcp 0 48 aHost.members.l:ssh someHost.
at:12681 ESTABLISHED
tcp6 0 0 [::]:ssh [::]:* LISTEN
```

Последний пример с *netstat* позволяет найти все TCP-соединения чисто с IPv4:

```
$ netstat -a -t -4
```

Мы здесь только прошлись по поверхности: опций в *Netstat* гораздо больше.

```
> legend('topright', c("Свободная память","Подкачка"), lty=1,
col=c("green", "red"), bty='n', cex=.75)
```

Зеленая линия в созданной диаграмме (рис. 4) — для первого столбца (объем свободной памяти), а красная линия — для второго столбца (использованная область подкачки). Как видите, показатель «использованной области подкачки» почти идеально постоянный, что очень хорошо по вышеупомянутым причинам.

Мониторинг ошибок страниц

Когда программа работает, она исполняется в ОЗУ и использует ОЗУ для выполнения разных своих задач. Если программа выдает запрос на память, кэшированную на диске, Linux возвращает ее в ОЗУ для использования программой. Подобным же образом, когда не хватает объема ОЗУ, часть ОЗУ перекачивается на диск до того момента, когда в ОЗУ высвободится требуемый объем. Поэтому Linux должен прекратить выполнение текущих процессов, чтобы копировать память с ОЗУ на диск и с диска в ОЗУ, и это именуется ошибкой страницы. Большое количество ошибок страниц является серьезным сигналом того, что с производительностью системы Linux что-то не так.

Есть масса способов мониторинга ошибок страниц; к сожалению, большинство из них требуют привилегий root, что может представлять опасность для системы. Простейший способ добыть информацию о деятельности, связанной с ошибкой страницы, не имея привилегий root — команда *vmstat*. Пример ее вывода:

```
$ vmstat
procs -----------memory---------- ---swap-- -----io---- -system--
----cpu----
r b swpd free buff cache si so bi bo in cs us sy id wa
0 0 27016 89532 76940 387048 0 0 28 8 3 1 2 0 97 0
```

Интересующие вас столбцы именуются si и so. Столбец si показывает объем памяти, подкачанной с диска, а столбец so показывает объем памяти, подкачанной на диск.

Следующий скрипт *Bash* извлекает искомые показатели:

```
$ cat pFaults.sh
#!/bin/bash
/usr/bin/vmstat | head -3 | tail -n +3 | awk {'print $7 « « $8'}
$ ./pFaults.sh
0 0
$ crontab -l
*/1 * * * * /home/mtsouk/bin/pFaults.sh >> ~/pageFaults.data
```

Вы можете попытаться визуализировать данные, сохраненные в **pageFaults.data**, самостоятельно — наш урок является лишь началом вашего путешествия в мир мониторинга и визуализации Linux. Есть куда больше инструментов и технологий, которые вы сможете изучить, так что продолжение следует! LXF

Cython: Python среди исходника

Cython — это компилятор Python на C, и ваш верный проводник **Филип Херрон** покажет, как ускорить код в 12 раз! Ваш пробег может отличаться...



Наш эксперт

Филип Херрон помешан на Python (в чем и сознается) после участия в Summer of Code Google три года подряд. Он работает в Instil Software (http://instil.co) в Белфасте.



Возможно, вы слышали о Cython. Обычно он всплывает в дискуссиях о Python и быстродействии. Cython — это язык программирования, который дополняет Python рядом ключевых слов и конструкций, чтобы воспользоваться «родными» операторами C. То есть, если вы напишете '1+2', Cython распознает здесь два целых числа и скомпилирует в C:

```
int result = 1 + 2;
```

Здесь есть интересные побочные эффекты, которые мы увидим позже на нашем уроке, но пока давайте думать, что Cython — это Python с возможностью использовать вещественные типы данных и даже набирать код C. Прежде всего, для установки Cython можно воспользоваться системными менеджерами пакетов, например, *apt-get*, в котором есть пакет Cython, или загрузить последний релиз с помощью *pip*.

```
$ pip install cython
```

Также нужно установить заголовочные файлы и библиотеки Python для разработчиков — для этого есть скрипт *python-config*. В Ubuntu это делается командой

```
sudo apt-get install python-dev
```

А в Fedora — командой

```
sudo yum install python-devel
```

Скорая помощь

Cython не поддерживает традиционный импорт кода, так как один файл Cython представляет собой целый модуль на Python. Но в главный файл с кодом можно добавлять **include** в стиле C/C++, например, **include morecode.pxi**.

Теперь все готово для использования Cython. Чтобы увидеть его в действии, рассмотрим в Python классическую функцию чисел Фибоначчи:

```
def fib(n):
  if n == 0:
    return 0
  elif n == 1:
    return 1
  else:
    return fib(n-1) + fib(n-2)
```

Этот код на Python вам, несомненно, знаком. Он возвращает n-й элемент последовательности Фибоначчи. Но теперь в Cython можно просто переписать объявление функции так:

```
cdef fib(int n):
  if n == 0:
    return 0
  elif n == 1:
    return 1
  else:
    return fib(n-1) + fib(n-2)
```

Как видите, этот код почти идентичен предыдущему, но здесь мы объявили функцию Cython **cdef** и объявили параметр **n** как целое число. С помощью этой информации Cython может оптимизировать код, так как мы знаем, что число **n** здесь точно целое, и нам знакома область видимости данных внутри функции, поэтому мы можем воспользоваться стеком C. Однако процесс компиляции и запуска этого кода чуть более сложен. Вместо того, чтобы просто запустить этот код в интерпретаторе Python, нужно запустить компилятор Cython и затем с помощью *gcc* скомпилировать код в исполняемый файл:

```
$ cython --embed fib2.pyx -o fib2.c
$ gcc -g -O2 -o fib2 fib2.c `python-config --includes --ldflags`
```

Сравним время выполнения обеих реализаций функции для первых 30 чисел Фибоначчи. Для этого добавим следующий код Python для вызова каждой функции, так как Cython прекрасно скомпилирует этот код!

```
result = []
for i in range (30):
  result.append(fib(i))
print result
```

Итак, сравниваем версии:

```
$ time ./fib1.py # Версия чисто на Python
real 0m0.537s
user 0m0.529s
sys 0m0.006s
$ time ./fib2 # Версия на Cython
real 0m0.045s
user 0m0.040s
sys 0m0.005s
```

Всё — модуль, все круто

В Cython один исходный файл представляет собой цельный модуль Python. При написании более крупных расширений Cython мы пользуемся директивами include, делая код более модульным. Проектируя собственный модуль расширения Cython, нужно подумать о том, как модуль будет запускаться. Как вы видите в примерах, мы передаем компилятору Cython аргумент embed. Он генерирует стандартный метод main для всей программы, так что выполнение начнется, как и можно ожидать, в коде Cython.

Но делать этого не обязательно, модуль можно сделать более похожим на обычный модуль Python — и мы видим это, когда не передаем этот аргумент и компилируем модуль как разделяемую библиотеку.

Однако для взаимодействия с этой разделяемой библиотекой нужно быть поаккуратнее с ее именем. Если файл с кодом Cython называется **MyModule.pyx**, то его нужно скомпилировать в **MyModule.c** и, наконец, с помощью *gcc* в **MyModule.so**.

В оболочке Python его можно импортировать простой командой **import MyModule**, так как этот модуль есть в вашем PYTHONPATH. Модуль также можно скомпилировать в разделяемую библиотеку или просто привязать к программе на C/C++ — но для этого потребуется инициализировать среду выполнения Python, а также модуль Python, причем до взаимодействия с кодом Cython, иначе программа «упадет». Вы можете увидеть это на примере с программой Flask.

Версия для Cython работает заметно быстрее — почти в 12 раз! Как мы сказали, тот факт, что мы пользуемся «родными» типами, означает: компилятор Cython знает, как полностью обойти среду выполнения Python для интерпретации этого кода, генерируя чистый код на C для выполнения задачи. Интересный побочный эффект тут вот какой: благодаря тому, что Cython генерирует код на C, мы можем обращаться к коду и структурам данных на C/C++ напрямую. А из кода на C снова обращаться напрямую к коду на Python!

Мы считаем, что это решает проблему с написанием сложных систем на C/C++ через JVM, где не хватает множества динамических абстракций; то есть вы можете перемежать программу на C/C++ кусками на Python, который позаботится о динамическом поведении для управления структурами данных. Даже при модульном тестировании кода на C/C++ Cython представляет собой прекрасный верхний уровень для имитации или настройки тестовой среды. Чтобы увидеть это в действии, встроим web-приложение Flask в приложение на C и передадим несколько внутренних структур данных к JSON. Я пользовался этим приемом в высокопроизводительных торговых системах — обращался с панели управления с web-интерфейсом к данным в разделяемой памяти, используя API на C++ вместо того, чтобы делать это на Python.

Этот процесс состоит из нескольких этапов; но сначала скомпилирум приложение *helloworld* с его домашней страницы http://flask.pocoo.org — разумеется, предварительно установив его:

```
$ pip install flask
```

Теперь перейдем к Hello world:

```
from flask import Flask
app = Flask(__name__)
@app.route("/")
def hello():
    return "Hello World!"
if __name__ == "__main__":
    app.run()
```

Как видите, Cython может компилировать код Python в C, но здесь он будет использовать среду выполнения Python. Для получения кода на C снова воспользуемся компилятором Cython, а затем с помощью *gcc* скомпилируем его в исполняемый файл:

```
cython -2 --embed app.pyx -o app.c
gcc -g -O2 -o app app.c `python-config --includes --ldflags`
```

Проясним параметры, использованные при вызове компилятора: **-2** означает «сгенерировать код для Python 2.x», а **--embed** означает, что нужно включить в него метод main. То есть в файле **app.pyx** будет метод main, и выполнение начнется с него. Если не указать эти параметры, придется запускать среду выполнения Python вручную и импортировать модуль... иначе программа «упадет»!

Представим, что мы хотим управлять с помощью C запуском сервера приложений. Тогда можно обернуть вызов **app.run()** в собственный метод, вызываемый из C. Для этого можно опять же воспользоваться функцией 'cdef', но с другим ключевым словом — 'public':

```
cdef public void startFlaskApp():
    app.run()
```

Тип возвращаемого значения — void, но важнее то, что ключевое слово 'public' заставит компилятор Cython создать заголовочный файл **app.h**, где создастся прототип этой функции. Поэтому данный код Cython можно вызывать прямо из C как любую функцию C, и все, что нам остается — подключить заголовочный файл, и вперед.

Так как наш метод main будет в программе на C, а для запуска web-приложения мы будем вызывать Cython, нужно вставить еще кое-какой стандартный кусок.

```
#include <Python.h>
int main (int argc, char **argv) {
    printf ("Initializing Python Runtime...\n");
    Py_Initialize ();
// Это перескок в Python
    printf ("Cleanup...\n");
    Py_Finalize ();
    return 0;
}
```

Поэтому внутри главной функции мы должны инициализировать среду выполнения Python и очистить ее. Между этими двумя вызовами можно вызывать код Python; в другом месте это повлечет ошибку сегментации. Помните, что поскольку мы »

› Запуск Cython'изированной версии программы с числами Фибоначчи и кодом в *Emacs* под ней.

объявили cdef для startFlaskApp как public, то можем включить после **Python.h** заголовочный файл **app.h**. И вызвать нашу функцию Cython напрямую:

```
// Обязательно сперва включите 'Python.h'!
#include <Python.h>
#include "app.h"
int main (int argc, char **argv) {
  printf ("Инициализируем Python Runtime...\n");
  Py_Initialize ();
  initapp(); // Эквивалент части 'import app' из 'app.h' в Python
  startFlaskApp(); // Вызываем функцию!
  printf ("Очистка...\n");
  Py_Finalize ();
  return 0;
}
```

Скомпилируем все это вместе:

```
$ cython -2 app.pyx -o app.c
$ gcc -g -O2 -o app main.c app.c `python-config –includes
--ldflags`
```

И запустим **./app**. Теперь у нас есть приложение Flask, которое работает из программы на C! Далее определим функцию, которая просто возвращает какие-то случайные данные в C-Struct. Например, для простоты, фиксированную строку и количество вызовов; представим, что это запись для какой-то торговой системы, так как принцип абсолютно тот же.

```
struct data {
  char name[10];
  int value;
};
```

Это определение нашей структуры, и теперь нам нужна только функция, которую можно вызвать и получить новую запись.

```
static struct data * record = NULL;
struct data * getData (void)
{
  static int callCount = 1;
  if (!record)
    record = malloc (sizeof (struct data));
  memset (record, 0, sizeof (*record));
  strncpy (record->name, "test", sizeof (record->name));
  record->value = callCount;
  callCount++;
  return record;
}
```

Эта функция просто заполняет запись, а в Python все это довольно элегантно обработается как объект JSON. Определение структуры, а также прототип функции для получения данных лучше всего поместить в заголовочный файл. Это важно, так как Cython, который компилирует все это в C, должен знать: нужно включить заголовочный файл в приложение на C, чтобы все успешно состыковалось.

```
#ifndef DATA_H
#define DATA_H
struct data {
  char name[10];
  int value;
};
extern struct data * getData (void);
#endif //DATA_H
```

В коде на Cython надо снова объявить этот заголовочный файл, чтобы компилятор Cython знал, как корректно обращаться с типами. Иначе компилятор Cython не узнает об этой структуре или функции, что приведет к ошибке.

```
cdef extern from "data.h":
  struct data:
    char name[10]
    int value
  data * getData ()
```

Как видите, код до боли знакомый; но писать его снова скучно. Есть способы (автоматическая генерация pxd — http://bit.ly/LXFapg) сгенерировать этот код за вас, но они еще не совсем доделаны. Затем нам нужна функция, в которую мы обернем вызов кода C и которая вернет словарь возвращаемой C-структуры.

```
cpdef getBackendData():
  cdef data * resp = getData ()
  return {'name': resp.name, 'value': resp.value}
```

В данной функции мы использовали **cpdef** — это ключевое слово означает, что функцию можно вызывать из кода на Python или на C, а в теле функции — использовать расширение Cython для употребления родных типов. Если вы собираетесь вызвать ее из кода на C, то возвращаемым типом всегда будет PyObject (внутренний тип объекта Python). В теле функции просто прямиком вызывается функция getData, которую мы определили в коде на C, чтобы превратить эту структуру в словарь Python. Наконец, чтобы объединить все это вместе, нужно изменить процедуру Flask на JSONify и обработать ответ!

```
from flask import Flask, jsonify
app = Flask ('myapp')
cpdef getBackendData():
  cdef data * resp = getData ()
  return {'name': resp.name, 'value': resp.value}
@app.route("/")
def index ():
  return jsonify (getBackendData ())
cdef public void startFlaskApp ():
```

› Запуск программы *Flask* и использование *curl* для доступа к ресурсу JSON.

```
redbrain@planet-derp: ~/workspace
{~/workspace/cython-article/example-flask} $ make
cython -2 flaskApp.pyx -o flaskApp.c
gcc -g -O2 -o example main.c flaskApp.c `python-config --ldflags --includes
{~/workspace/cython-article/example-flask} $ ./example
Initilizing Python Runtime...
Importing Flask App...
Starting Flask App...
 * Running on http://127.0.0.1:5000/
127.0.0.1 - - [16/Sep/2014 21:38:58] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [16/Sep/2014 21:39:00] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [16/Sep/2014 21:39:01] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [16/Sep/2014 21:39:03] "GET / HTTP/1.1" 200 -
127.0.0.1     [16/Sep/2014 21:39:04] "GET / HTTP/1.1" 200
127.0.0.1     [16/Sep/2014 21:39:05] "GET / HTTP/1.1" 200

{ } $ curl -X GET "http://localhost:5000"
{
  "name": "test",
  "value": 1
}{ } $ curl -X GET "http://localhost:5000"
{
  "name": "test",
  "value": 2
}{ } $ curl -X GET "http://localhost:5000"
{
  "name": "test",
  "value": 3
}{ } $ curl -X GET "http://localhost:5000"
{
  "name": "test",
  "value": 4
}{ } $ curl -X GET "http://localhost:5000"
{
  "name": "test",
```

» **Пропустили номер?** Узнайте на с. 108, как получить его прямо сейчас.

Альтернативы Cython

Подобной связи языков также можно достичь с помощью Swig. Достоинства Cython — в приведении типов и прямой реализации. Например, в Swig понадобилось бы написать множество функций-оберток для типов для выполнения приведений между разными языками, а также для обработки более сложных структур данных. Cython делает это за вас, так как он понимает типы! Это не значит, что Swig плох, но он слишком общий, поскольку поддерживает множество языков — Java, Go и т. д., и не может заранее включать стандартные куски для всех. Cython же предназначен исключительно для Python и может подготовить гораздо больше предварительных условий. Другие заметные проекты в этой сфере — Parakeet, который использует LLVM для компиляции кода, применяя для ускорения декораторы в коде Python, и Nuitka — это компилятор с Python на C++, который пытается статически типизировать вашу программу на Python во время компиляции, аналогично PyPy.

```
  app.run ()
```

Резюмируем. Пока у нас есть один файл на C с методом main для инициализации среды выполнения Python и запуска flaskApp. А также функция для получения данных, соответствующий заголовочный файл и файл Cython (***.pyx**), содержащий код приложения Flask для обработки данных. Скомпилировав все это, мы можем состряпать простой Makefile:

```
all:
   cython -2 flaskApp.pyx -o flaskApp.c
   gcc -g -O2 example main.c flaskApp.c `python-config --includes
--ldflags`
```

Запустим приложение:

```
$ ./example
Initialising Python Runtime...
Importing Flask App...
Starting Flask App...
  * Running on http://127.0.0.1:5000/
```

Для доступа к Flask воспользуемся *curl* — обратите внимание, как каждый запрос обновляет объект JSON.

```
$ curl -X GET "http://localhost:5000"
{
  "name": "test",
  "value": 1
}
$ curl -X GET "http://localhost:5000"
{
  «name»: «test»,
  «value»: 2
}
```

Вот и все — мы успешно вызвали код Python из C и код C из Cython! С этим можно сделать много больше — на некоторых сайтах я засек полезные ресурсы для классов C++. Поддержка C++ в Cython становится все лучше и лучше — вы даже можете определять шаблоны и перегружать операторы для классов, и Cython знает, как все это подхватить.

> «Мы смогли преспокойно писать код то на Cython, то на C++.»

У Cython есть несколько подводных камней, о которых следует знать заранее. Так как языки смешиваются, отладка может усложниться. Интерфейсы совершенно необходимо оставлять простыми, и обработка исключений Python на C++ может быть неуклюжей, так как исключения будут меняться в зависимости от того, с какого на какой язык вы переходите. У вас не получится волшебным образом скомпилировать весь модуль Python в разделяемую библиотеку, так как Cython не выполняет импорты. Для обработки нескольких файлов Cython нужно использовать директиву include, и это работает так же, как в C/C++ — включаемый код помещается в тело программы. Причина — в необходимости соответствовать модульной системе Python. Например, вы, возможно, заметили, что мы часто использовали **--embed** или сами определяли метод main в Cython, поэтому мы можем создать модуль

```
def myfunction ():
  return 'Hello World'
```

и скомпилировать его командами

```
cython -2 mycode.pyx -o mycode.c
gcc -g -O2 -shared -o mycode.so mycode.c `python-config
--includes --ldflags`
```

Теперь у нас есть разделяемая библиотека, и ее можно подключить точно так же, как и любой другой модуль Python:

```
>>> import mycode
>>> mycode.myFunction ()
'Hello World!'
```

Просто убедитесь, что имя разделяемой библиотеки совпадает с именем файла кода Cython с расширением **bar**. Cython на этом не заканчивается: у него есть кристально чистые интерфейсы для работы с кодом C++ включая классы и шаблоны, а что касается обработки исключений в Python и C++, то есть поддержка отличного отладчика *GDB* (GNU Debugger).

Автор этой статьи работал с большими многопроцессорными торговыми системами, активно использующими базы данных в разделяемой памяти, так что вся обработка происходит в памяти и незачем выполнять лишнюю работу, общаясь с базой данных. Недостаток такого подхода в том, что отладка и просмотр этих таблиц обычно затруднен; есть и другие проблемы, скажем, разрастание разделяемой памяти. Но с помощью Cython и Flask мы смогли по описанным здесь технологиям создать доску объявлений с web-интерфейсом, написав на Python и Angular. JS эпическую утилиту просмотра транзакций базы данных в реальном времени. Более того, вместо подключения к разделяемой памяти на Python мы просто воспользовались нашим API C++ и сделали все это прямо в Cython, то есть смогли преспокойно писать код то на Python, то на C++.

Это и правда было весьма впечатляюще; в больших финансовых учреждениях вы всегда получите лучший отклик, если при написании нового API или другого кода приветствуете расширение и повторное использование готового кода.

Эту методику также очень легко применить к другим областям, даже если дело доходит до написания новых программ на C/C++. Почему бы не написать обертки для логгера или парсера конфигурации Python и применить их в программах на C/C++, чтобы заново не изобретать велосипед для таких простых вещей? Если вы хотите посмотреть на одну из таких технологий, прочтите шедевр Филипа Херрона «Изучая Cython [Learning Cython]», который доступен в Amazon (см. также http://bit.ly/LearningCython и https://github.com/redbrain/cython-book). Хотя Cython первоначально предназначался для внедрения родных типов ради высокой производительности математических вычислений, он пригодится и в других сферах, а не только в науке. LXF

Mathematica Pi: Распознаем лицо

Осваиваем сложный мир распознавания образов с **Джонни Бидвеллом** — он научит вас угадывать с помощью Raspberry Pi, какие лица кому принадлежат.



Наш эксперт

Джонни Бидвелл чувствует себя малость неловко от того, что у него в компьютере так много фотографий команды **LXF**.



В **LXF190** на стр. 92 мы поместили для вас учебник по Pi-редакции программы *Mathematica*, вкладу Wolfram в Raspbian OS. А на этом уроке мы подробно изучим одну из особенностей Pi — поддержку модуля камеры Raspberry Pi, а также займемся простейшим определением и распознаванием лиц. Это жутко сложная тема, но не бойтесь — мы сведем все разговоры о матрицах ковариации и собственных пространствах к минимуму, а чтобы все это попробовать, вам потребуется только модуль камеры.

Существует два варианта модуля камеры: первый склонен к чернухе (почему и называется Pi NoIR) и распознаёт инфракрасное излучение, что позволяет ему видеть в темноте. Второй, стандартный модуль лучше подходит для более традиционной дневной фотографии. Каждый модуль стоит около £20, умеет снимать изображения с разрешением 5 Мпикс и записывать HD-видео, и пригоден для самых разнообразных домашних проектов по части видеонаблюдения. Модуль камеры подключается 15-контактным шлейфом к порту CSI, который расположен рядом с разъемом HDMI.

После подключения модуль камеры нужно активировать, что достаточно легко делается с помощью программы *raspi-config*. Перезагрузив систему, вы сможете делать снимки с командной строки простой командой

```
$ raspistill -o photo.png
```

Наряду с *raspistill* можно воспользоваться командой *raspiyuv* для получения «сырых» изображений (RAW), которые потом будут обработаны более детально.

Если все в порядке (в смысле, с камерой: с более общими экзистенциальными печалями мы вам не поможем), запустите *Mathematica* и введите такую команду для взаимодействия с властительницей душ... т.е. камерой:

```
In[1]:= img = DeviceRead["RaspiCam"]
```

Скажите «сы-ыыр»

По нажатию Shift+Enter эта команда выполнится, и ваше свежеснятое изображение появится в блокноте с префиксом Out[1]. Впредь мы не будем пользоваться подсказками и префиксами **In[]** и **Out[]**, иначе запутаемся. Если вам хочется работать с картинкой покрасивее, воспользуйтесь любым из доступных примеров в Интернете:

```
ExampleData[«TestImage»]
```

Для примера можно указать портрет своего очередного автомобиля, командой

```
img = ExampleData[{«TestImage»,»Tank»}]
```

Если надо сохранить изображение, это делается так:

```
Export["/home/pi/img.jpg", img]
```

Изображения в *Mathematica* хранятся в виде матриц, содержащих данные пикселей. Для изображений RGB каждый элемент матрицы по умолчанию представляет собой триаду из красного, зеленого и синего компонентов пикселя. Поскольку в вашем



› **Вот наша исходная галерея субъектов в розыске, в мрачной шкале серого. Будь они в *Papers, Please* [*Документы, пожалуйста* — компьютерная игра, — *прим. пер.*], вы бы впустили их к себе?**



› **Вот наше среднее лицо. Я имею в виду, усредненное.**

изображении пикселей довольно много, возьмем только квадрат размером 10 на 10 пикселей из его середины:

```
imgcrop = ImageCrop[img,{10,10}]
ImageData[imgcrop] // MatrixForm
```

Значение каждого компонента по умолчанию нормализуется в диапазон от 0 до 1. Мы можем перевести его в более привычный диапазон для байта (0–255), изменив последнюю команду:

```
ImageData[imgcrop, «Byte»] // MatrixForm
```

Почему бы вам не сфоткать нескольких друзей и (разумеется, с их разрешения) не занести их в базу данных распознавания лиц?

Mathematica в тематике лиц

В *Mathematica* масса встроенных методов для работы с изображениями, однако не все они портированы на Pi. В частности, своим отсутствием известна команда **FindFaces**, и очень жаль, так как с ней мы бы плавно перешли к следующему разделу. Эта функция возвращает удобный список координат контуров элементов изображения, напоминающих лица. Кстати, не было бы приятнее и безопаснее, если бы компьютеры [*Ред*.: Орудийные башни?] лучше распознавали людей?

Благодаря алгоритму распознавания собственных лиц [eigenface] мы легко сделаем именно это: имея в распоряжении некую начальную базу данных лиц, попробуем распознать человека, который есть в этой базе, по другой фотографии. Разумеется, этому подходу далеко до распознавания лиц человеческим мозгом — алгоритм не будет работать, если голова наклонена или лицо приняло новое выражение (отличное от выражений, которые уже есть в базе данных). Этой частью руководства мы обязаны записи в блоге Джереми Кана [Jeremy Kun] (http://bit.ly/Eigenfaces), в которой также можно найти ссылки на полный код.

Для создания нашей базы воспользуемся фотографиями «случайно» выбранных авторов из редакции **LXF**, а чтобы в Pi не иссякла память, возьмем миниатюры фотографий. Пусть наши портреты лежат в каталоге **~/faces** и имеют одинаковый размер. Мы можем импортировать и обесцветить их, создав массив с веселеньким именем **grayFaces** [*англ*. серые лица]:

```
files = Import["~/faces"];
faces = Map[Import["~/faces/" <> #] &, files];
grayFaces = Map[ColorConvert[#, "Grayscale"] &, faces];
```

Работа с изображениями в оттенках серого существенно упрощает жизнь. Помните, что в *Mathematica* они хранятся в виде матрицы (двумерного массива), каждый элемент которой означает яркость соответствующего пикселя. Стратегия собственных лиц опирается на математически простое, но психологически сложное понятие — понятие среднего лица. Не особо задумываясь о том, как такое личико объективно может выглядеть, мы можем создать среднее (или если хотите, усредненное) лицо, усреднив значения пикселей по нашей базе данных лиц.

```
meanImage = Image[
Apply[Plus, Map[ImageData, grayFaces]] /
Length[grayFaces]
]
```

Этот код просто складывает все матрицы grayFaces вместе и делит их на пять, а в результате мы получаем Среднестатистического Джо из *Linux Format* (на рис. слева). Измерив различие между каждым отдельно взятым лицом и средним лицом, мы получим нечто вроде базиса, от которого можно оттолкнуться при распознавании других лиц. Итак, создадим массив различий для лиц таким образом:

```
differenceFaces = Map[ImageSubtract[#, meanImage]&, grayFaces];
```

Более светлые области этого разностного массива — те, которые наиболее отклоняются от среднего лица. Темные области, напротив, те, в которых лицо близко к среднему. Алгоритм начинает работу с разворачивания матриц изображений в (довольно длинные) векторы; другими словами, навсегда забывает о разбиении на строки. Это легко сделать командой **Flatten** [*англ*. уплостить], а поскольку мы будем делать это часто, то напишем функцию, чтобы потом сэкономить себе немного времени. Также определим обратную функцию для преобразования вектора длины **nm** в матрицу с **n** строками и **m** столбцами:

```
imageToVector[img_] := Flatten[ImageData[img]];
vectorToImage[vec_, {n_, m_}] := Image[Partition[vec, m]];
```

Сформируем векторы разности лиц, развернув массив **differenceFaces**. Так как **differenceFaces** — трехмерный массив (пять изображений размером 180×200), мы получим матрицу размерностью 5×36000. Затем в дело вступает статистика, и мы формируем матрицу псевдоковариации для нашего вектора разницы лиц. Настоящая матрица ковариации имеет размер 36000×36000, а для нашего маленького компьютера это чересчур. Наше вычисление включает умножение векторов разницы лиц на их транспонированные версии, и оно не займет много времени.

```
differenceVectors = Map[imageToVector, differenceFaces];
```

»

Скорая помощь

Камера Pi NoIR нашла применение в изучении здоровья растений. При фотосинтезе растения испускают лучи, близкие по спектру к инфракрасным, что можно визуализировать, вычислив изображение различий нормализованного индекса вегетации (NDVI). Все об этом см. на сайте http://infragram.org.

```
In[8]:= differenceFaces = Map[ImageSubtract[#, meanImage] &, grayFaces]
In[9]:= differenceVectors = Map[imageToVector, differenceFaces];
In[10]:= Dimensions[differenceVectors]
Out[10]= {5, 36000}
```

› **Работая с разностями со средним изображением, из вычислений можно исключить общие элементы (темные области).**

❯ Линейная комбинация собственных лиц образует Криса Торнетта. Другая комбинация — доктора Брауна.

```
In[20]:= Map[rebuildFromEigenfaces[coefficients1, meanImage,
    eigenfaces, #] &, Range[4]]
```



```
In[23]:= Map[rebuildFromEigenfaces[coefficients2, meanImage,
    eigenfaces, #] &, Range[4]]

Out[23]=
```











```
covarianceMatrix = differenceVectors.Transpose[differenceVectors]
```

Все о ковариантности можно прочесть в Википедии, а пока достаточно понять, что она суммирует соотношения между парами пикселей в разностных изображениях. Вычислить значения собственных величин и собственных векторов для матрицы 5×5 нетрудно, а объяснение того, что это означает на самом деле, лучше оставить кому-нибудь другому:

```
eigenfaceSystem = Eigensystem[covarianceMatrix];
```

Собственные векторы — самый эффективный способ описания различий между фотографиями, хранящимися в нашей базе данных. Поскольку каждый собственный вектор может интерпретироваться как изображение, их можно назвать собственными лицами. Геометрически такому вектору соответствует направление в 36000-мерном пространстве, но визуально это сложновато представить. Составляя линейные комбинации собственных лиц, можно получать самые разнообразные новые лица.

```
displayEigenface[vec_] := vectorToImage[255 * vec,

In[15]:= Map[displayEigenface, eigenfaces]
```

❯ Эти (страшноватые) «собственные лица» — реальные архетипы для нашего процесса распознавания. “Eigen” в переводе с немецкого означает «собственный».

Чтобы увидеть собственные лица, сначала нужно умножить их на векторы разности, затем повторно нормализовать их и, наконец, представить их в виде изображений (с помощью нашей функции **vectorToImage**). Следующий код сделает это, и мы получим изображение, показанное ниже:

```
eigenfaces = Map[Normalize[Transpose[differenceVectors].#] &,
  eigenfaceSystem[[2]]];
displayEigenface[vec_] := vectorToImage[255*vec, dim];
Map[displayEigenface, eigenfaces]
```

Разбор лица

Изображение также можно разбить на уникальную сумму компонентов собственных лиц. Благодаря тому, что собственные лица образуют так называемый ортонормированный базис, это разбиение не представляет сложностей. Обратите внимание, что все вычисления выполняются на разностном изображении:

```
projectImageToFaceSpace[image_, meanImage_, eigenfaces_]:=
  Module[{imageVec, diffVec, meanVec},
  imageVec = imageToVector[image];
  meanVec = imageToVector[meanImage];
  diffVec = imageVec - meanVec;
  Map[(diffVec.#)&, eigenfaces]
 ];
```

Задав некоторые коэффициенты для каждого собственного лица, можно создать изображения путем добавления или вычитания различных комбинаций каждого собственного лица:

```
rebuildFromEigenfaces[coefficients_, meanImage_, eigenfaces_,
k_] :=
  vectorToImage[
  imageToVector[meanImage] +
  Apply[Plus, (coefficients*eigenfaces)[[;; k]]], dim];
```

» **Пропустили номер?** Узнайте на с. 108, как получить его прямо сейчас.

```
In[24]:= newGrayFace = ColorConvert[Import["~/nm2.jpg"], "Grayscale"]
Out[24]=
```



```
In[25]:= classifyNewFace[newGrayFace]
Out[25]=
```

{  ,  ,  , Classified}

› «Секретное» фото нашего босса в чуть лучшие времена доказывает, что система работает.

На рисунке показано постепенное улучшение путем добавления успешных собственных лиц к двум Крисам. Крис Торнетт распознается всего двумя собственными лицами, тогда как с Доктором все гораздо сложнее. Списки коэффициентов составляются простой командой:

```
coefficients1 =
  projectImageToFaceSpace[grayFaces[[1]], meanImage,
eigenfaces]
```

И чье же это лицо?

Мы создали все необходимое для того, чтобы разложить лицо на пять компонентов собственных лиц и чтобы выполнить обратное преобразование: собрать лицо по его спектру собственных лиц. Но чтобы в самом деле распознать лицо по фотографии, которой нет в нашей учебной базе данных, нужно чуть больше кода.

Параметр **threshold** [порог] можно изменить по желанию. Также, если у вас много учебных изображений, можно установить k немного меньшим, чем их общее количество — этот параметр определяет, на сколько собственных лиц будет осуществляться проекция.

```
Module[{trainingFacePoints, eigenfaceSubspace, k, threshold},
  k = 5;
  threshold = 25;
  eigenfaceSubspace = eigenfaces[[;; k]];
  trainingFacePoints =
  Map[projectImageToFaceSpace[#, meanImage,
eigenfaceSubspace] &,
  grayFaces];
  classifyNewFace[newFace_] :=
    Module[{projectedPoint, distances, index, min},
      projectedPoint =
        projectImageToFaceSpace[newFace, meanImage,
eigenfaceSubspace];
      distances =
      Map[EuclideanDistance[projectedPoint, #] &,
trainingFacePoints];
      min = Min[distances];
      index = Position[distances, min][[1, 1]];
      {grayFaces[[index]], newFace,
      rebuildFromEigenfaces[projectedPoint, meanImage,
        eigenfaceSubspace, k],
      If[min <= threshold, "Classified",
        «Unclassified: best guess provided»]}
    ];
  ];
```

В основе классификации лежит евклидово расстояние между проекцией нашего нового изображения и проекцией каждого из учебных изображений. Как вы видите из примера изображения, схема работает достаточно хорошо, чтобы распознать босса *Linux Format* в чуть более юном возрасте. Третье изображение — результат проекции на наши собственные лица, и оно достаточно близко к имеющемуся в базе данных, чтобы бдительный пограничник сделал на него стойку. Ну, а мы просто зафиксируем наличие соответствия. LXF

Что такое «собственное лицо»?

Метод eigenfaces — собственных лиц, возникший в 1987 году, представляет собой пример анализа главных компонентов. Точно так же, как в анализе Фурье волна описывается суммой синусов и косинусов, можно определить лицо как сумму лиц-компонентов, набор которых, однако, не может быть произвольным; желательно иметь их поменьше, но они должны быть достаточно разнообразными, чтобы покрыть все «пространство лиц». На языке математики, нам нужен ортонормальный базис пространства с малой размерностью.

Подход собственных лиц позволяет получить именно такой набор лиц-компонентов, причем сделать это без особо долгих или ресурсоемких вычислений. Собственные значения и собственные векторы — важнейшая часть линейной алгебры и тема этой статьи; метод позволяет быстро вычислить несколько собственных векторов матрицы ковариации (которая даже в нашем примере была бы огромной). Вместо того, чтобы связываться с этим чудищем, мы работаем с матрицей гораздо меньшего размера (в нашем случае 5×5) и в конце выполняем быстрое умножение, чтобы развернуть матрицы. Наш пример представляет собой довольно вырожденный случай, но если изображений, а следовательно, и собственных лиц в учебном примере более 200, это всё же некий перебор.

Flask: Создать монитор процесса

Филип Херрон преподаст практическую web-разработку тем, кто не говорит на Node.js, используя Python Flask, Angular.js и *PsUtil*.


Наш эксперт

Филип Херрон живет в Белфасте, Северная Ирландия. Он — большой любитель Python и участвовал в трех Программах Google Summer of Code по GCC и Python.

JavaScript получил буквально взрывное распространение в последние несколько лет, став повсеместным, и мы всё пытаемся делать в браузере как полноценной платформе для всех приложений. И на передовой этого процесса находится Node.js, реализация со стороны сервера JavaScript с открытым кодом. Он формирует общий язык для клиента и сервера и настолько популярен, что основные среды JavaScript не глядя предполагают: ваша серверная сторона — Node.js. Что вносит некоторую путаницу, если вы используете на серверной стороне Java или Python, например.

Я хочу показать вам, как Python может конкурировать с лучшими, быть более гибким и предлагать крутые web-приложения в среде, где вам будет удобно. Мы создадим монитор сервера с помощью Python Flask и *psUtil* на сервере. На клиентской стороне мы будем использовать Angular.js, который обеспечивает привязку данных, применяемую при создании по-настоящему динамичных сайтов, иногда именуемых «одностраничными» — например, Gmail.

Python Flask часто называют «микросредой», потому что он очень прост, а любая других функциональность обеспечивается использованием поверх него других библиотек. В самой сердцевине Flask предлагает HTTP-сервер разработки и переадресацию URL на особые функции. Давайте взглянем на 'helloworld':

```
from flask import Flask
app = Flask(__name__)
@app.route("/")
def hello():
  return "Hello World!"
if __name__ == "__main__":
  app.run()
```

Это приложение 'helloworld' от Flask просто возвращает 'Hello World', когда вы направляете свой браузер на http://localhost:5000. Вы также можете сделать это через *Curl*, что весьма удобно:

```
$ curl -X GET http://localhost:5000
```

Теперь, имея отправную точку, мы можем добавить маршрут для возвращения нужных нам данных. Итак, давайте напишем функцию, которая возвратит нам статистические данные о сервере:

```
import platform
hostStatistics = {
  'platform': platform.platform(),
  'hostname': platform.node(),
  'machine': platform.machine(),
  'version': platform.version(),
}
```

С помощью *platform*, встроенного модуля Python, мы можем получить простую информацию, например, имя хоста, платформа и тип машины. Поместив это все в простой словарь, мы используем помощник Flask, *jsonify*, который превратит словарь в настоящий ответ json HTTP:

```
https://registry.npmjs.org/qs/-/qs-1.2.2.tgz
https://registry.npmjs.org/mime-types/-/mime-types-1.0.2.tgz
https://registry.npmjs.org/caseless/-/caseless-0.6.0.tgz
https://registry.npmjs.org/stringstream/-/stringstream-0.0.4.tgz
https://registry.npmjs.org/got
https://registry.npmjs.org/registry-url
https://registry.npmjs.org/registry-url
https://registry.npmjs.org/got
https://registry.npmjs.org/registry-url/-/registry-url-0.1.1.tgz
https://registry.npmjs.org/got/-/got-0.3.0.tgz
https://registry.npmjs.org/ansi-regex
https://registry.npmjs.org/got/-/got-0.3.0.tgz
https://registry.npmjs.org/registry-url/-/registry-url-0.1.1.tgz
https://registry.npmjs.org/ansi-regex
https://registry.npmjs.org/ansi-regex/-/ansi-regex-0.1.0.tgz
https://registry.npmjs.org/mute-stream
https://registry.npmjs.org/ansi-regex/-/ansi-regex-0.1.0.tgz
https://registry.npmjs.org/mute-stream
```

› Здесь мы устанавливаем *Bower* через *npm*.

```
#!/usr/bin/env python
from flask import Flask
from flask import jsonify
app = Flask(__name__, static_folder=sfolder)
@app.route("/api/hostinfo")
def getHostInfo():
  return jsonify(
    {
      'platform': platform.platform(),
      'hostname': platform.node(),
      'machine': platform.machine(),
      'version': platform.version(),
    }
if __name__ == '__main__':
  app.run(debug=True)
```

А если мы это запустим и выполним вызов HTTP через *Curl* —

```
$ python server.py
$ curl -X GET "http://localhost:5000/api/hostinfo"
{
  "hostname": "planet-derp",
  "machine": "i686",
  "platform": "Linux-3.13.0-32-generic-i686-with-ubuntu-14.04-trusty",
  "version": "#57-Ubuntu SMP Tue Jul 15 03:51:12 UTC 2014"
}
```

Обратите внимание на **@app.route(«/api/hostinfo»)** — эта аннотация помощника сообщает Flask, что когда вы направляете свой HTTP-клиент на http://localhost:5000/api/hostinfo, функция выполняется, и возвращается ваш результат. Вы также заметите, что использовано **jsonify**. Это функция помощника Flask, которая берет словарь и создает для вас ответ объекта json. Таким образом, вам не нужно настраивать заголовки HTTP и сериализировать данные.

Теперь у нас есть простой вызов rest API для получения базовой информации о сервере, и нужно настроить сайт, чтобы ее видеть. Для этого создадим файл **index.html** и будем обслуживать его из Flask как web-страницу:

```
$ mkdir www
$ emacs index.html
<html>
  <header>
    <title>This is title</title>
  </header>
  <body>
    Hello world
  </body>
</html>
```

С этим **index.html** внутри папки **www** наш код будет чудесно размещен вне нашего кода сервера Python. Нам следует только сообщить серверу Flask, как обслуживать этот и другие активы из данной директории:

```
sfolder = os.path.join(os.path.dirname(os.path.abspath (__file__)), 'www')
app = Flask(__name__, static_folder=sfolder)
@app.route("/")
def index():
  return app.send_static_file('index.html')
```

Надо кое-что запомнить. Когда вы направляете свой браузер на сервер, он теперь будет возвращать HTML-файл, и мы также создали объект приложения Flask с новым параметром. Чтобы Flask обслуживал статические файлы из вложенной папки **www**, нам нужно сообщить ему, где искать эти файлы. Поэтому мы используем модуль **os**, чтобы получить текущую директорию файла Python, и ищем рядом с ним папку **www**. Вы можете установить собственные правила, или просто оставить ее без параметра, и тогда она по умолчанию будет искать файлы в текущей рабочей директории. Это хорошо, но наличие HTML внутри собственной папки — это просто замечательно и очень аккуратно. Поэтому теперь, запустив свое web-приложение

```
$ python server.py
```

и направив свой браузер на этот URL, вы увидите свой HTML-файл helloworld. Однако нам также нужны соответствующие зависимости JavaScript. Для этого мы воспользуемся *Bower* — предварительно нужно установить *npm* из Node.js:

```
# ubuntu
$ sudo apt-get install npm
# fedora
$ sudo yum install npm
```

Затем установим *Bower*:

```
$ sudo npm install -g bower
```

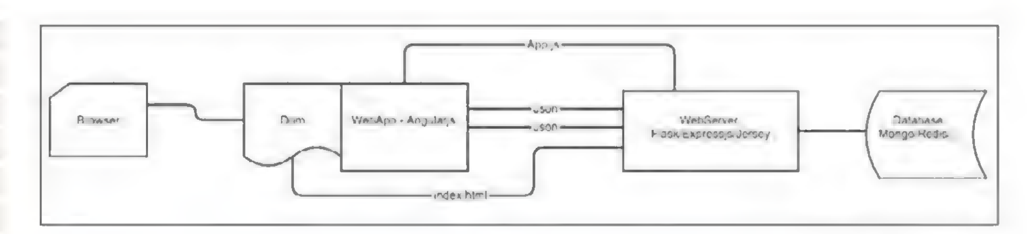
Примечание: я обнаружил, что после установки *Bower* двоичный скрипт, который будет в **usr/local/bin/bower**, имеет строку **#!/usr/bin/env node**, и вам может понадобиться изменить эту строку в зависимости от вашей инсталляции. Для самого последнего Ubuntu следует изменить эту строку на **#!/usr/bin/env nodejs**.

Теперь надо создать **.bowerrc** и **bower.json** внутри директории вашего проекта, чтобы определить, куда должны направляться зависимости в нашем проекте и какие зависимости нам нужны, соответственно. Создайте файл **.bowerrc** в корне вашей директории проекта:

```
$ emacs .bowerrc
{
  "directory": "www/js/lib",
  "interactive": false
}
```

Определим требуемые нам зависимости в **bower.json**, тоже в корне директории проекта.

```
$ emacs bower.json
{
  "name": "Flask-Article",
  "private": true,
  "dependencies": {
```



› Здесь показана общая архитектура встраивания Angular.js в web-страницу.

```
    "requirejs": "latest",
    «angular»: «1.2.x»
  }
}
```

Bower требует имени нашего проекта, но его маркировка в качестве частного и не зарегистрированного как публичное приложения означает, что имя роли не играет. Финальная настройка — это список ключевых значений зависимостей и соответствующая версия. Мы здесь будем использовать две зависимости. Первая — **requirejs**, и она очень удобна в больших приложениях JavaScript: она означает, что мы можем определить все зависимости JavaScript внутри одного файла, так что не нужно постоянно обновлять сам наш HTML-файл для работы с большим количеством активов, и т. д., что помогает поддерживать порядок. Второе — базовый пакет Angular.js. Итак, давайте рассмотрим всё это в действии. Сначала установим зависимости:

```
$ bower install
```

Теперь сообщим Flask, как обслуживать эти зависимости для web-приложения. Поскольку всё вложено в **www**, нам нужна трассировка Flask, которая возьмет параметр пути в зависимости внутри папки **www**. Итак, если мы сделаем http://localhost:5000/index.html, это сработает, а вот http://localhost:5000/js/lib/angular.js вернет JavaScript. Мы можем это сделать так:

```
@app.route(«/<path:pathparameter>»)
def statics(pathparameter):
  return app.send_static_file(pathparameter)
```

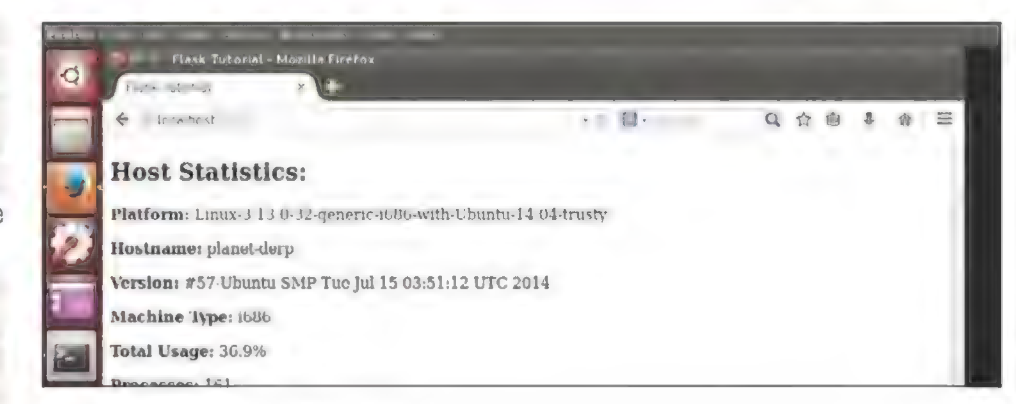
Эта трассировка будет обслуживать статические файлы и зависимости внутри нашей папки **www**, чтобы клиент запускал их корректно. Синтаксис **<path:name>** будет соответствовать полному пути и встраивать путь в качестве параметра ключевого слова «name» в функцию. Например, http://localhost:5000/css/myapp.css передаст в качестве параметра **/css/myapp.css**, и функция помощника flask, **send_static_file**, будет искать эти файлы в папке **static_folder**, которую мы указали при инициализации приложения.

Теперь, если вы запустите сервер и направите свой браузер на http://localhost:5000/js/lib/requirejs/requirejs.js, он вернет JavaScript, и мы сможем соответственно работать с путем для других зависимостей!

Давайте затем настроим наше приложение Angular и добавим вызовов rest сервера для получения большего количества данных. Возвращаемся к нашему **index.html**:

```
$ emacs www/index.html
<!DOCTYPE html>
<html lang="en">
```

»



› Панель управления web-страницы.

```
  <head>
    <title>Flask Tutorial</title>
    <script src="/js/lib/requirejs/require.js" data-main="/js/app.
js"></script>
  </head>
  <body>
  </body>
</html>
```

Это будет наша web-страница. Пока что она ничего не отображает, но вы увидите строку **requirejs** — она велит *requirejs* загрузить зависимости и код в файл **/js/app.js**. Так создадим же **app.js**:

```
$ emacs www/js/app.js
require.config({
  paths: {
    angular: '/js/lib/angular/angular',
  },
  shim: {
    'angular': {
      exports: 'angular'
    }
  },
  deps: ['app']
});
```

Это настройка *requirejs*. В наши цели не входит вдаваться в подробности насчет *requirejs* и загрузки модулей, но я полагаю, что это хороший способ загрузить все зависимости в одну строку вместо множества тэгов <script> внутри HTML-файла. В нашем случае нужен Angular.js, но надо описать **shim**, чтобы загрузить Angular должным образом. Вам не нужно делать это со всеми зависимостями — например, для **jquery** вы можете просто указать другой путь, без **shim**. Подробное объяснение см. на http://requirejs.org/docs/api.html. Сначала это может показаться усложнением, но когда вам нужно будет интегрировать Bootstrap и другие плагины Angular, на самом деле все упростится — все изменения находятся внутри одного файла, и не надо менять файлы HTML. Конечная директива **deps** указывает другую зависимость, **app**, которая будет нашим приложением JavaScript!

```
define('app', ["angular"], function(angular)
{
  var app = angular.module("StatsApp", []);
  angular.bootstrap(document, ['StatsApp']);
  return app;
});
```

Это наша базовая инициализация проекта Angular. Функция **define** — из **requirejs**, что дает коду оболочку в виде зависимости, которую можно использовать повторно. Она берет имя зависимости (в нашем случае — 'app'), список зависимостей (здесь просто 'angular' из нашей настройки *requirejs*), и функцию кода, надстраиваемую как зависимость. **function** объявляет наше приложение Angular и выполняет самозагрузку приложения — мы используем это вместо директивы **angular ng-app**, которую нам применить нельзя, поскольку мы загружаем модули асинхронно и директива может быть недоступна вовремя для обработки в домене. Мы просто делаем это напрямую, когда все уже готово.

Bootstrap и Angular

Bootstrap.js — это популярная среда для создания отличных сайтов, хотя в чистом виде Bootstrap и Angular.js при совместной работе иногда конфликтуют. Проекты вроде http://angularui.github.io/bootstrap облегчают весь этот процесс. Вы можете просто добавить его в качестве еще одной зависимости в ваш bower.json, и вот он у вас есть! Добавочное преимущество в том, что мы уже настроили *requirejs*, а значит, вам не нужно добавлять эти противные тэги <script> в свой HTML — вы можете хранить все JavaScript в файлах js. Кроме того, *requirejs* подходит не только для работы со средами JavaScript — он также может работать в качестве механизма ввода зависимости, подобного Objection, Spring MVC или Google Guice. Это дает вам приятную возможность распространить свой код в отдельные области, что облегчает добавление в ваше приложение макетирования и модульных тестов.

› **Здесь мы устанавливаем зависимости JavaScript.**

Ну вот, все связано, и мы можем приступить к созданию приложения. Помните, как мы реализовали API json для получения информации хоста?

```
$ curl -X GET "http://localhost:5000/api/hostinfo"
```

Мы можем динамически отобразить эти данные в нашем web-приложении! Понадобится определить контроллер приложения в Angular:

```
define('app', [«angular»], function(angular)
{
  var app = angular.module("StatsApp", []);
  app.controller('app', function($scope, $http) {
    $http.get('/api/hostinfo').success(function(data) {
      $scope.info = data
    })
  })
  angular.bootstrap(document, ['StatsApp']);
  return app;
});
```

Итак, вы видите, что мы добавили контроллер Angular под названием 'app', который принимает функцию. Angular встроит эти зависимости **$scope** и **$http**. Первая из них относится к DOM — HTML, известному, как привязка данных: когда **$scope** обновляется, **dom** автоматически обновляется для нас. **$http** обеспечивает поддержку *jquery $.ajax* в Angular, так что мы можем осуществлять вызовы сервера для получения данных. Затем мы можем вызвать **$http.get** для запроса HTTP GET на **/api/hostinfo**, и это автоматически поставит ответ в зависимость от удачного завершения объекта *json*. Поэтому мы можем просто выполнить **$scope.info = data**. Так что информация об атрибутах **$scope** — это **hostinfo** с сервера.

Возвращаясь к нашему **index.html**, мы можем заявить некоторые директивы Angular для отображения этих данных в DOM:

```
<body ng-controller="app">
  <h2>Host:</h2>
  <p><b>Platform:</b> {{info.Platform}}</p>
  <p><b>Hostname:</b> {{info.hostname}}</p>
  <p><b>Version:</b> {{info.version}}</p>
  <p><b>Machine:</b> {{info.machine}}</p>
</body>
```

Мы объявили атрибут **ng-controller=»app»**, который велит Angular использовать контроллер 'app', объявленный нами ранее. Далее нужно сослаться на данные, которые вы хотите отобразить в DOM, с помощью директив **{{}}** или *ng-bind*. Использование

скобок более распространено, однако и у *ng-bind* есть свое применение. И вот, как по волшебству, Angular.js обработает все и разместит все это для вас. Но заметьте, мы не ссылаемся на **$scope** — у нас есть доступ ко всем атрибутам, которые мы добавили в **$scope**.

А теперь давайте добавим парочку крутых функций с помощью *python-psutil*. Это удобный модуль для получения информации хоста приятным и простым способом. Например:

```
$ sudo pip install psutil
$ python
>>> import psutil
>>> psutil.cpu_count()
...
>>> psutil.cpu_times_percent().user
...
>>> psutil.virtual_memory()
...
```

Вы видите, как много можно найти, приложив столь незначительные усилия: если подправить наш вызов **hostinfo**, внедрив это, мы легко сможем скомандовать Angular выполнить отображение:

```
@app.route("/api/hostinfo")
def getHostInfo():
  return jsonify(
    {
      'Platform': platform.platform(),
      'hostname': platform.node(),
      'machine': platform.machine(),
      'version': platform.version(),
      'cores': psutil.cpu_count(),
      'usage': psutil.cpu_times_percent().user,
      'memory_total': psutil.virtual_memory().total,
      'memory_used': psutil.virtual_memory().used,
      'disk_total': psutil.disk_usage('/').total,
      'disk_free': psutil.disk_usage('/').used,
      'timestamp': datetime.datetime.now().isoformat(),
      'process': len(psutil.pids())
    }
  )
```

И теперь все, что нам нужно сделать — это отредактировать наш **index.html** для работы в этих дополнительных областях:

```
    <p><b>Platform:</b> {{info.Platform}}</p>
    <p><b>Hostname:</b> {{info.hostname}}</p>
    <p><b>Version:</b> {{info.version}}</p>
    <p><b>Machine Type:</b> {{info.machine}}</p>
    <p><b>Total Usage:</b> {{info.usage}}%</p>
    <p><b>Processes:</b> {{info.process}}</p>
    <p><b>Cores:</b> {{info.cores}}</p>
    <p><b>Total Memory:</b> {{info.memory_
total/1024/1024/1024}}G</p>
```

Angular берет на себя самую тяжелую работу, заставляя сервер вызывать и обрабатывать связывание данных для нас. Нам остается только объявить, что мы хотим видеть.

Давайте-ка перейдем на следующий уровень. С помощью *psUtil* мы можем получить статистику использования каждого CPU, и затем дать Angular команду делать это в реальном времени:

```
@app.route("/api/hoststats")
def getHostStats():
  return jsonify(
    {
      'total_usage': psutil.cpu_percent(percpu=False),
      'cpu_usages': psutil.cpu_percent(percpu=True),
      'memory_total': psutil.virtual_memory().total,
      'memory_used': psutil.virtual_memory().used
    }
  )
```

Этот вызов вернет использование CPU и памяти по каждому запросу на **/api/hoststats**, и такая информация всегда будет свежей и новой. Затем вы можете определить новый контроллер Angular или добавить к существующему контроллеру. (Ради простоты мы добавим его к существующему контроллеру приложений, хотя на практике неплохо бы, чтобы они были по отдельности). Сработает и простое добавления нового запроса **$http.get** для этого нового ресурса, но в идеале нам надо, чтобы он автоматически обновлялся и мы могли бы воспользоваться преимуществом другой директивы Angular, **$interval**:

```
app.controller('app', function($scope, $interval, $http) {
  var hostInfo = function() {
    $http.get('/api/hostinfo').success(function(data) {
      $scope.info = data
    })
  }
  var hostStats = function() {
    $http.get('/api/hoststats').success(function(data) {
      $scope.stats = data
    })
  }
  $interval(hostInfo, 2000)
  $interval(hostStats, 1000)
})
```

Заметьте, что раз мы добавили зависимость **$interval**, определение контроллера изменилось. Далее мы возвращаем каждый вызов ресурса в его собственную функцию, чтобы все выглядело аккуратнее. Вы видите, что происходит: **$interval** берет функцию и время в миллисекундах, требуемое для запуска функции в этом интервале.

Чтобы все это отображалось автоматически, нужно обновить наш HTML для отображения новых данных:

```
    <h2>Host Usage:</h2>
    <p><b>Memory Used:</b> {{stats.memory_
used/1024/1024/1024}}G</p>
    <p ng-repeat="core in stats.cpu_usages"><b>Core {{$index}}
</b> {{core}}%</p>
```

Для динамического отображения статистики по каждому CPU, которая является списком значений, нам нужно воспользоваться директивой *ng-repeat*, чтобы перебрать список статистики CPU. По большей части мы делаем 'for i in stats.cpu_usages: ...', а используя **$index**, мы ссылаемся на количественный индекс в списке, тогда как Core будет ссылаться на настоящие значения. Теперь, когда вы откроете свою web-страницу, все будет динамически загружаться туда для вас и автоматически обновляться!

Вы можете отправиться отсюда хоть во все стороны! Интеграция Gruntfile для запуска задач будет весьма удобна для настоящих проектов, где у вас имеются активы и множество CSS и JavaScript. Интеграция *uglify* или *less* сделает ваше приложение готовым к производству, с соответствующей минификацией, интегрированной в вашу сборку. Кроме того, вы уже настроили *requirejs* и *Bower* на добавление новых зависимостей.

Надеюсь, это вдохновило вас обратиться в web-разработке к Python. Многие считают, что в Node.js многовато ограничений на стороне сервера, а вот Java или Python уже занимаются этой работой долгие годы! LXF

Скорая помощь

Для использования приложений Flask я предпочитаю использовать обратный прокси-сервер *nginx* и запускать несколько экземпляров сервера Flask, настраивая **app.run(port=1234)** так, чтобы каждый из них мог использовать уникальный порт. Пока ваше приложение обеспечивает сохранность данных в движке, это очень хорошо работает. Другие методы использования Flask см. на http://flask.pocoo.org/snippets/category/deployment.


```
  "memory_total": 772341760,
  "memory_used": 617795584,
  "process": 162,
  "timestamp": "2014-10-13T21:06:55.939615",
  "usage": 27.5,
  "version": "#57-Ubuntu SMP Tue Jul 15 03:51:12 UTC 2014"
}
```


› Случайный пример запуска *Curl* на сервере rest.

Ответы

Есть вопрос по открытому ПО? Пишите нам по адресу answers@linuxformat.ru, и мы найдем ответ.

1 Скрытый дискоед

Я пользовался Linux Mint на своем ПК более трех лет. Сначала у меня была версия 11, затем около года назад я перешел на 13-ю. Она работает хорошо, но у меня проблема с корневым разделом, которой я не понимаю. Если заглянуть в Control Centre > System Information > Filesystems [Центр управления > Информация о системе > Файловые системы], я вижу, что раздел заполнен на 94,4% (свободно 1,3 ГБ из 23,2 ГБ). Но если открыть *Baobab*, *Filelight* и т.д., я просто не вижу, куда исчезло все это место. Если заглянуть в свойства отдельных каталогов в /, в общей сложности получается 8,3 ГБ. Почему же в информации о системе указано, что свободно всего 1,3 из 23,2 ГБ? Мой каталог /home находится на отдельном разделе.

Fraxinus, с форумов

Был ли каталог **/home** на отдельном разделе в прошлой системе? Подозреваю, что нет. Это может означать, что файлы прежнего каталога **/home** все еще находятся в корневом разделе, но невидимы. В Linux каждая файловая система монтируется в «точку монтирования» — каталог, в котором затем размещается содержимое файловой системы. Обычно за точку монтирования принимается пустой каталог, но это не обязательно. При монтировании файловой системы в каталог, который уже содержит файлы, эти файлы становятся невидимыми, заменяясь содержимым новой файловой системы. Вы можете не видеть эти файлы, но прежнее содержимое каталога занимает то же место на диске, и при размонтировании файловой системы и освобождении точки монтирования они появляются снова.

Если Вам досаждают все эти старые файлы, но они недоступны глазу, то как их удалить? Один из вариантов — загрузиться с Live CD, смонтировать корневую систему (но не **/home**) и изучить ее содержимое. Однако есть более умный прием, который избавляет от необходимости перезагрузки: это связанное монтирование. Откройте терминал и выполните следующие команды:

```
sudo mkdir -p /mnt/root
sudo mount --bind / /mnt/root
```

Они смонтируют каталог / в **/mnt/tmp**. Вы увидите одно и то же содержимое в обоих местах, и любые изменения в одном месте отразятся на другом; но имеется одно важное различие. Все, что монтируется в каталог в /, не будет монтироваться в соответствующий связанный каталог. Такие каталоги, как **proc** и **sys** в **/mnt/root**, будут пустыми, как и **mnt**, **media**, **home** и подобные им каталоги. Запустив на **/mnt/root** *Baobab* (анализатор использования диска) или *Filelight*, Вы увидите, что на самом деле содержится в корневой файловой системе. Также это можно сделать в терминале командой **du**:

```
sudo du -sch /mnt/root/* | sort -h
```

Подозреваю, что в **/mnt/root/home** найдется масса старых файлов. Вы, наверное, захотите их удалить (они остались с предыдущей установки), но, возможно, стоит сначала перенести их во временный подкаталог в **/home**. Это освободит место в / и позволит Вам просмотреть и удалить нужные файлы по собственному усмотрению, не перезагружая систему.



› Анализатор использования диска покажет, куда деваются байты, но он не может учесть файлы, скрытые при монтировании поверх них другой файловой системы.

2 Разрушение тайком

Мой жесткий диск начал барахлить и выдавать ошибки чтения. Я заменил его и скопировал с него все файлы, кроме нескольких, которые не скопировались из-за ошибок чтения. Понимаю, что эти файлы уже не вернуть, но как я могу быть уверен, что с остальными все нормально? Не были ли они повреждены из-за разрушения жесткого диска? Можно ли их проверить, чтобы потом не получить неприятный сюрприз?

Эндрю Райт [Andrew Wright]

Если у Вас не одна из файловых систем последнего поколения (например, ZFS или btrfs), которые автоматически проверяют контрольные суммы всех файлов и их целостность, универсального решения нет. Если у Вас есть резервные копии файлов, можете сравнить файлы с резервными копиями. Скопируйте или извлеките резервные копии куда-нибудь, например, на внешний диск, перейдите в каталог командой **cd** и выполните команду **md5deep** таким образом:

```
md5deep -r . >all.md5
```

Эта команда сформирует свертки MD5 для каждого файла в текущем каталоге и каталогах нижнего уровня и сохранит их все в файл. Скопируйте этот файл в свой домашний каталог, перейдите в него командой **cd** и наберите

```
md5deep -x all.md5 -r .
```

Параметр **-x** означает, что нужно сравнить файлы с контрольными суммами в файле **all.md5** и вывести только те файлы, контрольная сумма для которых не совпадает. Потом Вы сможете решить, поврежден ли файл или просто изменился с момента создания копии, но трудов при этом все равно гораздо меньше.

Если резервных копий нет, кое-что, кроме обещания делать их в будущем, все-таки можно сделать. Некоторые файлы умеют проверять себя сами. Например, любой файл TAR, сжатый или нет, можно проверить с помощью параметра **-t**. По существу, эта команда распаковывает содержимое файла TAR, но не сохраняет его. Команда

```
tar -tf sometarball.tar.gz >/dev/null
```

не выведет ничего, если с файлом все нормально. Для поиска и такой проверки всех файлов TAR можно воспользоваться командой **find**:

```
find -name '*.tar' -or -name '*.tar.*' -exec tar -tf {} + >/dev/null
```

Команда **find** найдет все файлы, соответствующие любому из шаблонов имен, и запустит для них **tar -tf**, перенаправив вывод в **/dev/null**. В окне терминала будут показаны только сообщения об ошибках, поэтому отсутствие новостей — хорошая новость. Фигурные скобки **{}** в команде будут заменены списком файлов, соответствующих условиям поиска. Плюс **+** означает «добавить как можно больше файлов», поэтому если у Вас три архива, команда окажется такой:

```
tar -tf file1.tar file2.tar.gz file3.tar.bz2
```

Если заменить **+** на **;**, **find** будет запускаться отдельно для каждого найденного файла:

```
tar -tf file1.tar
tar -tf file2.tar.gz
tar -tf file3.tar.bz2
```

Похожим образом командой **jpeginfo** можно проверить файлы JPEG. Как и **md5deep**, она скорее всего не установлена по умолчанию, но уж точно есть в репозиториях Вашего дистрибутива. **jpeginfo** с параметром **--check** проверит целостность всех переданных ей JPEG-файлов. Для поиска имен файлов, передаваемых команде, опять же используется **find**:

```
find -iname '*.jpg' -exec jpeginfo {} + | grep -v \ [OK\]
```

На сей раз мы указали параметр **-iname**, чтобы учесть и **.jpg**, и **.JPG**. С помощью *grep* мы исключаем файлы с отметкой [OK], и остаются только предупреждения и ошибки.

3 RDP RPi — гр-рр!

В **У меня на Raspberry Pi установлен Raspbian с *xrdp*. Каждый раз, когда я подключаюсь к нему клиентом удаленного рабочего стола в Mate 17, открываю терминал и набираю**

```
emacs &
```

я получаю следующее сообщение об ошибке:

```
Xlib: extension "RANDR" missing on display ":10.0".
```

***Emacs* запускается, но он не похож на себя самого при запуске напрямую с Raspberry Pi — теперь он черно-белый и без цветной заставки.**
Я попробовал поискать решение этой проблемы в Google и на различных форумах, но так и не смог ничего найти.
Роб Мандин [Rob Mundin]

О Здесь есть несколько ложных отвлекающих моментов. Хотя Вы сталкиваетесь с этой проблемой при запуске *Emacs* (и приятно видеть, как кто-то пользуется нормальным текстовым редактором, а не всеми этими *vi*), на самом деле это сообщение будет появляться при запуске почти любой программы, использующей X. Во-вторых, это сообщение просто предупреждает Вас о том, что Ваш сервер *VNC*, используемый RDP, не реализует расширение RANDR X. RANDR используется для управления корневым окном дисплея, например, при изменении размера рабочего стола. Для удаленных соединений это обычно не требуется. Это не сообщение об ошибке, и оно не влияет на работу других программ. Да, внешний вид *Emacs* при запуске по RDP отличается от обычного — наверное, потому, что он определяет, что запущен через удаленный рабочий стол, и отображает не такой красивый текстовый начальный экран. При этом никакой потери цвета нет — попробуйте загрузить скрипт на Python, и Вы увидите цветовое выделение. *Emacs* отображается в монохромном режиме потому, что отображает монохромную страницу, это никак не связано с отсутствующим расширением.

Для удаленного запуска одной программы не обязательно запускать полный рабочий стол. Вы уже в терминале, поэтому можете запустить удаленный сеанс оболочки в локальном терминале через SSH. Если запустить SSH с параметром **-Y** —

```
ssh -Y pi@address
```

то будет включено перенаправление X. Затем можно либо запустить программу из той удаленной оболочки, либо запустить ее командой **ssh**:

```
ssh pi@address emacs
```

Emacs или любая другая программа, которую Вы хотите запустить, откроет окно на рабочем столе. Вы увидите, что если запустить *Emacs* таким образом, он будет цветным.

4 Неродной язык

В **Я перехожу с Windows XP на Linux, но в мире Linux приходится говорить на иностранном языке. Как научиться понимать язык Linux?**
Ноэль Стюарт [Noel Stuart]

О Всегда приятно видеть, что кто-то отваживается перейти в другую ОС; но делать первые шаги всегда непросто. Linux, безусловно, отличается от Windows (иначе зачем огород городить?), и у него, как Вы обнаружили, применяется собственная терминология. Хорошая новость в том, что существует огромное сообщество пользователей Linux, все члены которого когда-то были новичками (даже если не все они »

Терминалы и суперпользователи

Мы часто предлагаем в качестве решения проблемы ввести те или иные команды в терминале. Хотя обычно то же самое можно сделать с помощью графических утилит дистрибутива, такие решения будут слишком конкретными (будут зависеть от дистрибутива). Команды в терминале более гибкие и — самое главное — ими можно пользоваться во всех дистрибутивах. Команды настройки системы часто нужно выполнять от имени суперпользователя, называемого также root. Существует два основных способа это делать, в зависимости от используемого дистрибутива. Во многих дистрибутивах, особенно в Ubuntu и его производных, перед командой можно написать **sudo** — при этом будет запрошен пароль пользователя, и ему будут предоставлены привилегии root только на время выполнения команды. В других дистрибутивах применяется команда **su**, для использования которой требуется ввести пароль root и которая предоставляет полный доступ root до того момента, пока вы не наберете **logout**. Если в вашем дистрибутиве используется **su**, запустите ее один раз и выполняйте любые заданные команды без предшествующей **sudo**.



Коротко про...

Файловые ссылки

В Linux широко используются файловые ссылки, в основном символические (или мягкие). Файловая ссылка — это способ для одного и того же файла (или каталога) появляться в двух или более местах или под различными именами. Можно было бы просто скопировать файл в каждое местоположение, но здесь есть несколько недостатков: во-первых, трата дискового пространства, а во-вторых, если основная версия файла изменится, копия останется устаревшей.

В файловых системах Linux есть средства доступа к одному и тому же файлу или каталогу из нескольких мест — ссылки. Существует два типа ссылок: жесткие и символические (symlink). С технической точки зрения каждый файл представляет собой жесткую ссылку, и это ссылка с имени файла на структуру данных на диске с его содержимым. Когда мы говорим о жестких ссылках, мы имеем в виду дополнительные ссылки, поэтому у файла есть несколько имен. У жестких ссылок есть некоторые ограничения: главное — они применяются только к файлам, а не к каталогам, и все ссылки должны ссылаться на ту же файловую систему.

Символические ссылки гораздо более гибкие, и в типичной системе Linux встретить их можно повсюду. Чтобы их увидеть, добавьте параметр **-l** к **ls**; они также отображаются в большинстве файловых менеджеров. Классический пример можно найти в каталоге **/usr/src**, где **/usr/src/linux** — символическая ссылка на исходный код текущего ядра, например, *linux-2.6.15*. Так можно иметь исходные коды нескольких установленных ядер, и исходники для текущей версии всегда можно найти через **/usr/src/linux**. В **/etc/rc.d** тоже много символических ссылок.

Символическую ссылку можно создать командой

```
ln -s /path/to/file /path/to/link
```

Один или оба пути могут быть относительными.



› С помощью RDP легко запустить удаленный рабочий стол, но это не лучший способ запустить текстовый редактор.

готовы это признать), поэтому всегда найдутся желающие Вам помочь.

Первое, с чего нужно начать, находится прямо под рукой — в разделе "New to Linux" на **LXFDVD** можно найти объяснение многих основных моментов: откройте файл **index.html** с DVD в браузере и щелкните по пингвину в левом верхнем углу. Одной из первых задач станет выбор дистрибутива. Дистрибутив — это один из тех самых терминов Linux; он означает «полноценная ОС на основе Linux». Дистрибутивов на свете много; одни больше подходят для новичков, другие — меньше. Здесь играет роль не только содержимое дистрибутива, но и сообщество вокруг него. Дистрибутивы вроде Ubuntu или Linux Mint хорошо подходят новичкам и имеют массу ресурсов, способных помочь неопытным пользователям.

Я бы посоветовал Вам выбрать какой-нибудь дистрибутив и остаться в нем, а не прыгать с одного дистрибутива на другой. Тогда у Вас будет время привыкнуть к терминологии и возможностям именно этого дистрибутива и получить помощь от его сообщества. У всех дистрибутивов имеются форумы и списки рассылки, через которые пользователи и разработчики могут помогать друг другу, и на многих из них предусмотрены разделы специально для новичков, в которых можно не стесняясь задать свой вопрос.

Открою Вам секрет: между самими дистрибутивами нет большой разницы — в конце концов, все это Linux. А значит, стоит Вам освоиться с одним дистрибутивом, и полученные знания сослужат Вам хорошую службу, когда Вы решите попробовать другой дистрибутив. Если не получить этих начальных знаний, сколько бы Вы ни прыгали с одного дистрибутива на другой, это не поможет.

Есть и более общие ресурсы вроде http://linuxquestions.org; Вы можете также попробовать наши форумы (www.linuxformat.com/forums), где новые пользователи всегда встречают радушный прием.

Linux не более сложен, чем Windows — он просто другой. Выберите себе дистрибутив, пользуйтесь доступными ресурсами сообщества, и скоро все это обретет смысл. Пройдет немного времени, и Вы окажетесь по другую сторону преграды, помогая новым пользователям на форумах выбранного дистрибутива.

Обновление ядра LMDE

В У меня Linux Mint Debian Edition на 32-битном настольном компьютере, и я хотел бы знать, как обновить ядро до последней стабильной версии, если это возможно...

Hampshire_Hog1, с форумов

О Кроме желания получить ядро более свежей версии, причин такого обновления может быть две: поддержка устройств и обновления безопасности. В последнем случае для дистрибутива будет выпущено обновление. На данный момент, более нового пакета ядра для LMDE по сравнению с тем, который был поставлен установщиком 201403, нет. Если Вам нужно более свежее ядро, его придется скомпилировать из исходников, что не так уж сложно. В LMDE есть пакет *kernel-package*, который еще больше упрощает эту задачу. Это набор программ для создания пакетов ядра Debian из стандартного исходного кода ядра. Сначала зайдите на http://kernel.org и загрузите последний архив с исходными кодами, на момент написания статьи — 3.16.2. Затем нужно установить пакеты *kernel-package* и *linux-source*. Первый пакет — сборщик ядра, второй — исходный код текущего ядра; сам код нам не нужен, а нужен конфигурационный файл из этого пакета. Откройте терминал и переключитесь на пользователя root, командой

```
sudo bash --login
```

Перейдите в каталог с исходными кодами ядра и распакуйте загруженный архив:

```
cd /usr/src
tar -xf ~user/Downloads/linux-3.16.2.tar.xz
ln -s linux-3.16.2 linux
```

Последняя команда не обязательна, но некоторые установщики модулей ожидают, что исходный код ядра будет находиться в **/usr/src/linux**. Пакет *linux-source* установит более старый исходный код, по-прежнему в виде архива, который можно игнорировать, и набор конфигурационных файлов в *linux-config-3.11*. Выберите тот, что подходит для Вашего ядра (если не уверены, запустите **uname -a**), и распакуйте его в соответствующее место каталога с новыми исходниками:

```
xzcat linux-config-3.11/config.i386_none_486.xz
>linux/.config
```

Теперь можно запустить **make-kpkg** из каталога с исходниками:

```
cd /usr/src/linux
make-kpkg buildpackage
```

Установленный конфигурационный файл предназначен для более старой версии кода, поэтому первое, что нужно сделать — запустить **make oldconfig** для его обновления. Эта команда будет

Помогите нам помочь вам

Ежемесячно мы получаем несколько писем, на которые не в состоянии ответить, поскольку проблема описана в них недостаточно полно. Чтобы дать вам наилучший ответ, нам необходимо знать как можно больше.

Если у вас появляется сообщение об ошибке, приведите его точный текст и опишите конкретные условия, когда оно появляется. При возникновении проблемы с устройствами перечислите нам все установленные устройства.

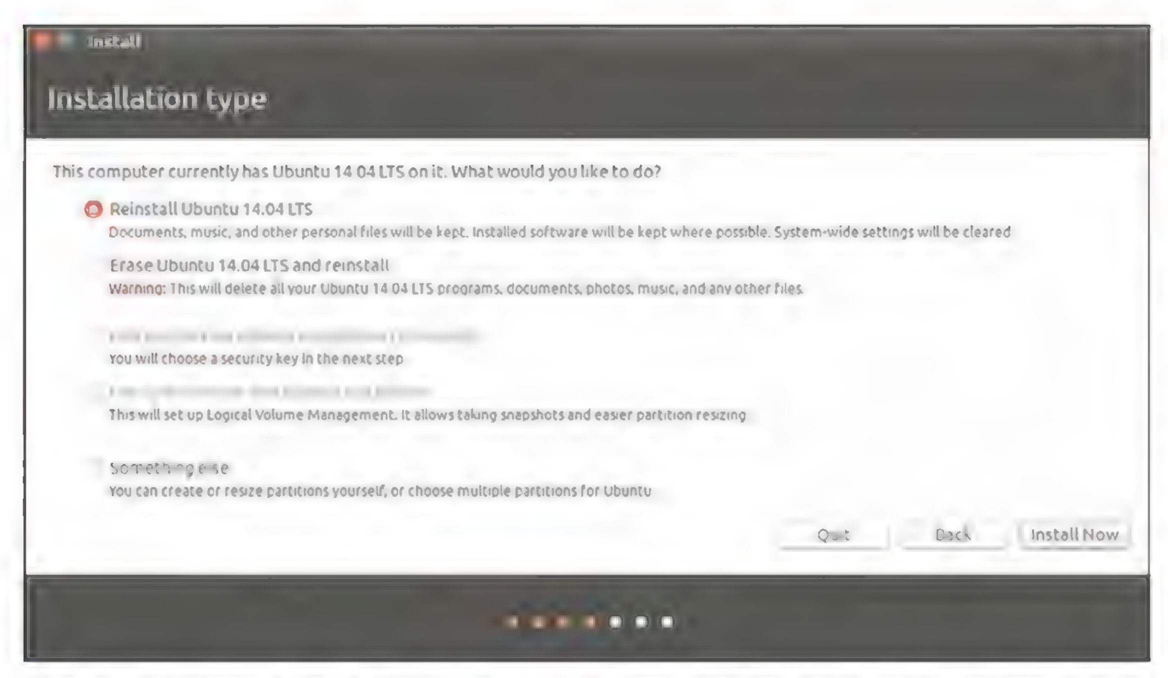
Если Linux уже запущен, можете применить для этого отличную программу *Hardinfo* (http://hardinfo.berlios.de/) — она сохранит подробную информацию об устройствах и о состоянии системы в HTML-файле, который вы сможете приложить к своему письму.

Не уступающий в удобстве альтернативный вариант — *lshw* (http://ezix.org/project/wiki/HardwareLiSter). Одна из указанных программ непременно должна быть включена в ваш дистрибутив (а иногда и обе).

Если вы не хотите или не можете их установить, выполните следующие команды в терминале от имени root и приложите файл **system.txt** к письму. Это здорово поможет диагностике.

```
uname -a >system.txt
lspci >>system.txt
lspci -vv >>system.txt
```





› **При выборе варианта Reinstall в Ubuntu Remix от *Linux Format*, да и в любом другом дистрибутиве Ubuntu ОС будет установлена, но ваши личные файлы это не затронет.**

запрашивать значения новых параметров ядра с вариантами по умолчанию. Можно нажать ?, чтобы узнать больше, или Enter, чтобы принять вариант по умолчанию, который подойдет в большинстве случаев. Версии 3.11 и 3.16 значительно отличаются, и большинство из новых параметров касаются нового оборудования, поэтому смотрите, не появится ли Вашего. Команда **make-kpkg** скомпилирует ядро и соберет его в Deb-пакет в **/usr/src**. Его можно установить следующей командой:

```
dpkg -i ../linux-image-3.16.2_3.16.2-10.00.Custom_i386.deb
```

Точный номер версии будет отличаться. Затем Вы сможете перезагрузиться в свое новое ядро, зная, что если что-то пойдет не так, в меню загрузки еще сохранилось старое.

6 Обновление на Remix

В У меня Ubuntu 14.04, установленная на 64-битном ноутбуке HP 250 (с отключенным UEFI), и мне хотелось бы воспользоваться DVD, записанным из ISO-образа Remix из LXF185. Могу ли я обновить систему с нового DVD, а не переустанавливать ее, как предлагается в процессе установки Remix?
Cushie, с форумов

О Да, можете. По существу, установщик Ubuntu каждый раз форматирует диск и распаковывает на него огромный архив, поэтому каждая установка — это на самом деле переустановка. При обновлении установщик действует стандартно, но сохраняет содержимое определенных каталогов, таких как **/home**, и все Ваши файлы и настройки остаются нетронутыми. Это работает, даже если **home** не находится на отдельном разделе.

Установщик определяет, что у Вас уже установлена та же версия дистрибутива, и предлагает выбрать между переустановкой и обновлением. Однако, хотя при этом персональные данные сохраняются, системные настройки не сохраняются. Поэтому перед установкой можно сделать копию каталога **/etc**, чтобы потом скопировать настройки обратно. То же относится к **/var/lib**, если у Вас имеются сервисы, которые хранят в нем данные — например, базы данных или даже программа *BitTorrent*.

При загрузке с DVD неважно, какие параметры рабочего стола Вы выберете. Они влияют только на то, какой рабочий стол будет загружен по умолчанию. Независимо от этого, установщик скопирует их все на Ваш жесткий диск. LXF



Часто задаваемые вопросы

Что такое руткит?

Руткит — набор программ, устанавливаемый взломщиком чтобы скрыть свое присутствие на компьютере. Он включает, например, модифицированные версии **ls** и **ps**, которые не показывают файлов и процессов, принадлежащих взломщику. Чтобы снизить подозрения, программы обычно прячутся, как трояны. Название объясняется тем, что эти программы устанавливаются и могут запускаться от имени пользователя root. Пораженный компьютер принято называть "rooted".

› Какой ущерб системе обычно наносит руткит?

Сам по себе не такой большой. Его задача — дать злоумышленнику путь к вашему компьютеру так, чтобы у вас не возникло подозрений. Для этого он в основном изменяет программы для отслеживания состояния системы таким образом, чтобы они не замечали ничего необычного. Этим руткиты отличаются от троянов.

› Что такое троян?

Троянский конь — часто сокращается до «троян» — вредоносная программа, замаскированная под нормальную. Названием он обязан коню, с помощью которого греки проникли в осаждаемую ими Трою — отсюда фраза «бойтесь данайцев, дары приносящих, и сайтов со взломанными программами». Пример трояна (к счастью, не представляющего большой проблемы для пользователей Linux) — программы с рекламой. К ним же относятся и другие программы, которые «звонят домой» без вашего ведома и согласия.

› Имеет ли это какое-то отношение к музыкальным компакт-дискам?

Ну разве что последнее ПО с DRM от Sony описывается как руткит, хотя и для Windows. Тот факт, что оно устанавливается такой известной компанией на такое количество компьютеров без ведома и согласия пользователей, делает ее самым откровенным (на данный момент) руткитом.

› Как узнать, поражена ли система руткитом?

Существуют различные программы для проверки системы на признаки наличия руткита. Они работают двумя основными способами: путем поиска известных последовательностей кода руткитов и путем сравнения контрольных сумм различных файлов с контрольными суммами этих файлов в нетронутой системе. Из-за последнего факта важно установить и запустить программу как можно раньше после установки системы, чтобы она записала контрольные суммы для «чистой» системы. Две таких программы — *Rootkit Hunter* с www.rootkit.nl и *chkrootkit* с www.chkrootkit.org. Их нужно запускать регулярно, желательно из *cron*. Хорошо, если компьютер будет подключен к Интернету, чтобы они автоматически обновляли свои базы данных известных руткитов.

› Вылечат ли систему эти программы?

Нет. После установки руткита на компьютер система серьезно скомпрометирована. Руткит помогает взломщику замести следы своих действий, поэтому их действия могли затронуть и любые другие файлы. Единственный безопасный выход — сделать резервную копию своих данных и конфигурационных файлов и переустановить систему.

LXFHotPicks



Александр Толстой
надевает ушанку и идет охотиться на лучшие программы с открытым кодом, чтобы возвести их на пьедестал и дать возможность всем на них полюбоваться.

QMplay2 » Rosa ImageWriter » PDFSaM » Rodent Core » KXStitch » KEncFS » I-Nex » Blobby Volley 2 » Caesaria » Otter Browser » ModemManager

Медиа-плейер

QMplay2

Версия: 14.12.28 **Сайт:** http://zaps166.sourceforge.net

Это очередной медиа-плейер, способный работать почти со всеми видами аудио- и видеофайлов. Однако у *QMplay2* есть еще и ряд выдающихся функций, которые его отличают; ну и он приятнее в работе: это один из редких плейеров, позволяющих легко сортировать ваш плей-лист, не закрывая основного отображения видео. Свойство вроде и очевидное, но, например, в *VLC* отличный редактор плей-листа и отличное управление воспроизведением, и хотя вы можете переключаться между ними, оба элемента одновременно видеть нельзя. Правда, такие приложения, как *MPlayer* в Gnome, позволяют включить боковую панель с плей-листом — но не перетаскивать в нем пункты. Среди популярных медиа-плейеров Linux только *Totem* сумел объединить эти функции, но недавние новые версии *Totem* (начиная с 3.12) избавились от отдельной панели плей-листа — возможно, из-за перегруженности.

Итак, *QMplay2* вполне может оказаться медиа-плейером вашей мечты. Плейер предлагает одновременное отображение видеопанели и визуализацию плей-листа, и позволяет индивидуально настраивать весь свой вид по личному вкусу. Несмотря на свое название, *QMplay2* не является интерфейсом *MPlayer*. На самом деле, он использует для поддержки медиа-форматов *FFMpeg*, заворачивая его в *Qt4* (и, возможно, *Qt5*, если вам хватит отваги скомпилировать *QMplay2* из исходника) GUI с широким выбором настроек.



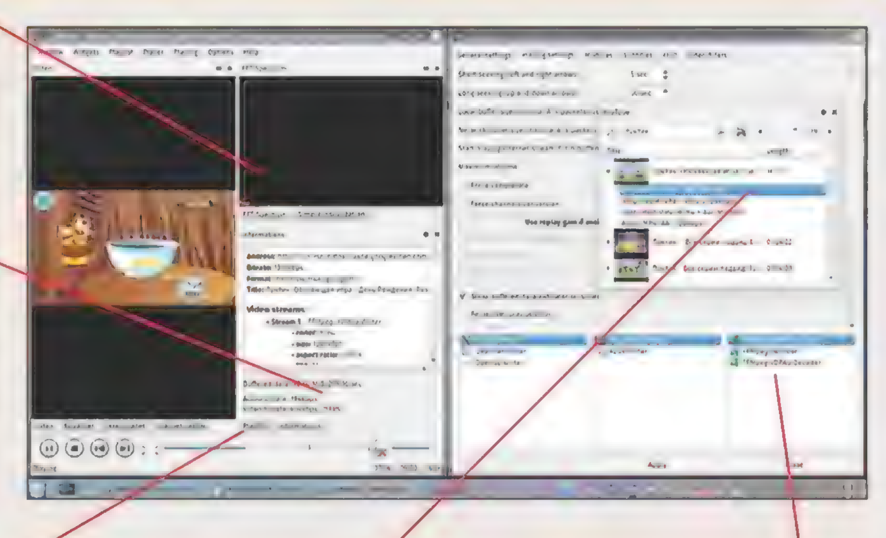
› Количество кнопок управления и вкладок в *QMplay2* весьма велико, что превращает его в отличный вариант для профессионального рабочего стола.

> «Несмотря на название, QMplay2 не является интерфейсом MPlayer.»

Выдающаяся его функция — интуитивный метод настройки вывода аудио и видео, который размещается в окне свойств плейера (чтобы его открыть, нажмите Ctrl+O). Очень легко настроиться на ускоренный вывод видео для работающих на VAAPI систем Intel Graphics, а пользователи Nvidia и Radeon, вероятно, смогут воспользоваться VDPAU.

Панель плей-листа способна принимать не только локальные файлы: она столь же непринужденно справляется с видео с YouTube и прочих URL потокового вещания. Плейер всегда восстанавливает предыдущий плей-лист при запуске, и всегда можно вернуться к ранее воспроизведенным элементам, не обращаясь к меню Recent или Favourite. Воспроизведение и поиск внутри видео YouTube работают чудесно, и если у вас достаточно быстрое соединение, вы не заметите разницы между локальным видео и воспроизведением потока.

Исследуем интерфейс QMplay2



Визуализатор аудио
Смотрите ли вы фильм или слушаете любимый трек, этот визуализатор с двумя опциями может оказаться весьма полезен...

Live-метаданные
Очень удобно для удаленного потока: виден объем находящихся на данный момент в буфере данных и их эквивалент в секундах, а также аудио live.

Сохраняющийся плей-лист
QMplay2 хранит ранее воспроизведенные элементы сколько вы хотите, и вы всегда сможете снова и снова наслаждаться ими.

Инструмент YouTube
Панель результатов поиска по YouTube позволяет мгновенно выбирать нужное качество воспроизведения.

Настройка приоритетов
Это, вероятно, самый удобный способ настроить желаемый приоритет вывода — просто перетаскивайте элементы!

Live USB flash

Rosa ImageWriter

Версия: 2.4 **Сайт:** http://bit.ly/ZgurWL

Оптические лазерные диски стремительно уходят из нашего мира: их вытесняют флэш-диски USB и облачное хранение. Хотя вы можете решить переместить свои драгоценные файлы в онлайн-сервис, но вам все равно придется иметь дело с физическим носителем при установке ОС (по крайней мере, пока). Подготовка устройства массового хранения USB (флэшки, карты памяти и т. д.), запись на него ISO вашего любимого дистрибутива Linux, и, наконец, получение загружаемого Linux может показаться несложным делом, особенно для гиков, поднаторевших в синтаксисе **dd**. Однако для простого смертного всё это по-прежнему остается задачей уровня эксперта. Хотя многие наслышаны про *UNetbootin*, и в мире Ubuntu к нему привыкли, не каждый ISO можно успешно записать на USB. Одни предназначены только для CD/DVD, другим нужна дополнительная настройка для поддержки EFI, а третьи предлагают нестандартное деление на разделы.

Rosa ImageWriter является частью Rosa Linux, который, в свою очередь, является продолжением дистрибутива Mandriva Linux. Несколько лет назад команда Rosa ответвила *Imagewriter* из SUSE Studio — скромный и очень простой в использовании инструмент для записи ISO-образов на устройства USB. Ответвление было хорошо оптимизировано и расширено; они избавились от кода C# и .NET соответственно в версиях Linux и Windows, и полностью переписали инструмент на C++. Интерфейс также был портирован в среду *Qt5*, сохранив при этом изначальную визуальную простоту.

В конечном итоге получился инструмент, который постоянно обновляет список имеющихся носителей USB, отображает свои отметки и размеры и поддерживает выбор ISO — через классическое диалоговое окно открытия файлов или перетаскивание файла на окно инструмента. *Rosa ImageWriter* распространяется в виде tar-архива исходника и набора двоичных пакетов для 32- и 64-битного Linux, Windows 7/8 и Mac OS X. Вместе с пакетами для конкретных дистрибутивов имеются статические сборки для всех платформ, поэтому вы можете подготовить свое устройство USB с ISO из любой современной системы Linux, даже из проприетарных чужаков. В наших тестах *Rosa ImageWriter* правильно обработал все производные Ubuntu и Debian, все виды дистрибутивов на базе RPM, образы FreeBSD и многое другое.



› Еще одна небольшая утилита *Qt5* с отлично разработанными функциями достойна пристального изучения.

> «Оживите устройство USB через ISO из любой современной системы.»

Инструмент разбиения и слияния PDF

PDFSaM

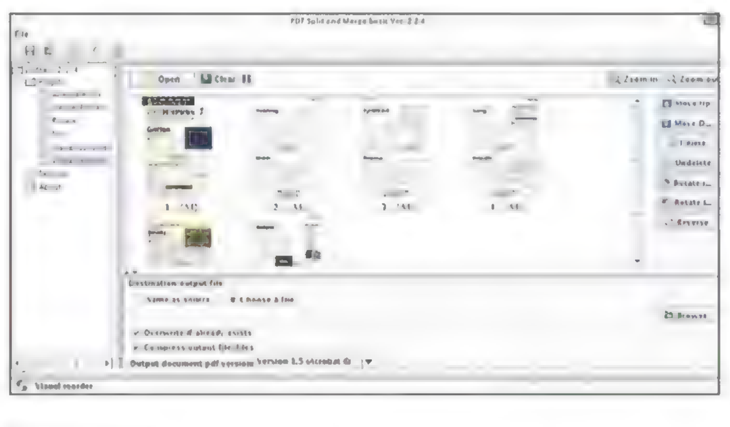
Версия: 2.2.4 **Сайт:** www.pdfsam.org

Вероятно, самым распространенным форматом взаимообмена файлов является PDF. В отличие от **.docx** или даже **.odt**, он все еще остается самым доступным, так как почти каждая ОС предлагает стандартный инструмент для чтения этого формата. Linux тоже не отстает: его приложения *Evince* и *Okular* — превосходные программы, и как минимум одна из них всегда идет в любом основном дистрибутиве, который вы выберете. Хотя PDF был разработан Adobe как целевой формат, то есть не предназначенный для модифицирования, в реальной жизни часто требуются некоторые изменения, например, повороты, кадрирование, слияние и выбор страниц и т. д.

Инструмент *PDF Split and Merge* (для друзей — *PDFSaM*) вырос из простенького приложения, разработанного для разрезания и разворота файлов PDF, в настоящий швейцарский нож в области работы с PDF, способный выполнять с PDF множество трюков. Это независимая от платформы и ОС программа, написанная на Java Swing, которая требует рабочей среды Java или OpenJDK (из-за чего, следует признать, интерфейс выглядит немного старомодным).

Распакуйте tar-архив PDFSaM и запустите **java -jar pdfsam-2.2.4.jar** в его директории. Интерфейс довольно необычен — по крайней мере, сначала. Вы думали, что сможете просматривать файлы PDF и применять к ним нужные действия, однако логика здесь обратная. *PDFSaM* — это, по сути, браузер плагинов модификации PDF: вы выбираете, что намерены сделать, и применяете свой PDF к этому плагину. На данный момент доступно шесть плагинов: Alternate mix (перемещает страницы между файлами PDF), Merge/Extract, Rotate, Split, Visual document composer и Visual reorder. Два последних плагина обеспечивают предпросмотр страниц PDF и их модификацию в режиме live, а остальные создают модифицированные файлы PDF в директории, указанной плагину. Некоторые опции, например, перезапись исходного файла или указание внутренней версии PDF, можно контролировать.

PDFSaM будет хорошим товарищем для тех, кто часто работает с файлами PDF или кому нужен кросс-платформенный инструмент для пакетной обработки файлов. В наших тестах он был вполне стабилен даже под солидной нагрузкой множества увесистых файлов. Реорганизация и перемещение страниц из одного файла в другой тоже работала у нас безупречно.



› Инструмент не слишком красив, но понятен и богат функциями.

> «PDFSaM — это, по сути, браузер плагинов модификации PDF.»

Менеджер файлов

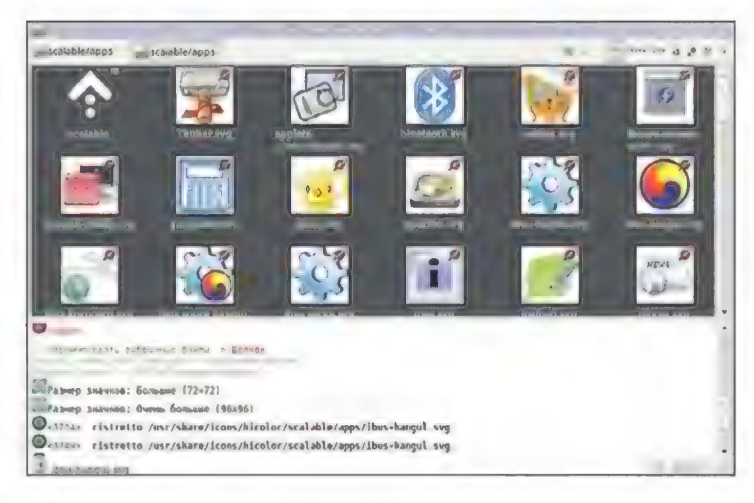
Rodent Core

Версия: 5.2.9 **Сайт:** http://xffm.org

Помимо тяжеловесных сред рабочего стола вроде KDE SC или Cinnamon, есть немало компактных и быстрых рабочих столов, которые всегда находят преданных пользователей — в основном владельцев старых ПК; но порой даже хозяева современных мощных монстров предпочитают не транжирить ресурсы системы, обходясь быстрыми программами попроще. Rodent Applications — типичная подборка небольших удобных утилит. У одного из его ключевых приложений, менеджера файлов *Rodent Core* (он же *xffm*), недавно вышел новый релиз. Это легковесное приложение *GTK2*, основанное на плагинах, с достойным набором функций для повседневного использования. *Rodent Core* выглядит и работает очень необычно, особенно если вы привыкли к классическим менеджерам файлов вроде *Dolphin* или *Nemo*. Он заменяет ваш текущий рабочий стол своим собственным, и все это выглядит наподобие графической мини-оболочки.

Главное окно *Rodent* по умолчанию настроено на отображение квадратных плиток файлов и директорий. Большая часть графики, PDF и даже текстовые файлы поддерживаются встроенной программой создания миниатюр, так что вы вряд ли заблудитесь, хотя парадигма *Rodent* весьма необычна. Для открытия файлов и выполнения других базовых действий *Rodent* использует многочисленные внешние утилиты, многие из которых стандартны в Linux. Просмотр с правами администратора использует **sudo**, данные автозаполнения текстового поля хранятся в *Bash*, а программа автомонтирования файловой системы является комбинацией **/etc/fstab** и интерфейса *FUSE*. Знакомясь с *Rodent*, вы наверняка заметите

> «Заменяет ваш рабочий стол своим, наподобие мини-оболочки.»



› Индивидуально скроенный менеджер файлов, несколько жестче контролирующий ваш рабочий стол.

уникальные функции: безопасное удаление файлов, простое шифрование файла с помощью *Bcrypt* и множество других удобных инструментов, например, интерфейсы для *diff* и *ssh*. В низу экрана *Rodent* расположена оболочка, где большинство действий с файлами дублируются соответствующими командами оболочки.

Решив попробовать менеджер файлов *Rodent Core*, позаботьтесь о системе, которую вы используете. Двоичные пакеты существуют для openSUSE, Gentoo, Ubuntu и FreeBSD, и хотя исходник доступен, как любое ПО GPL, самостоятельная компиляция *Rodent* может вызвать затруднения; но если у вас хватит пороху, вы получите невероятно быстрый менеджер файлов с низким содержанием жира.

Редактор узоров для вышивки крестиком

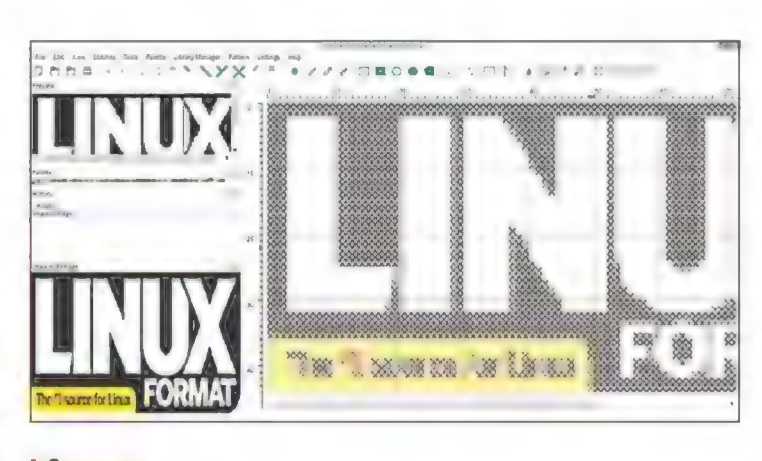
KXStitch

Версия: 1.2.0 **Сайт:** http://bit.ly/1s9wyXz

Хотите растиражировать рукодельный шедевр вашей бабушки, но не знаете, как описать рисунок вышивальной машине? Вот вам прекрасное решение. *KXStitch* — забавная программа, позволяющая трансформировать любой растровый графический файл в рисунок вышивки крестиком, применить редактирование вручную и создать инструкцию по выполнению вышивки. *KXStitch* использует собственный внутренний формат файлов для хранения схем вышивок (KXS), и чтобы использовать с этим приложением фото или чертеж, его надо сначала импортировать через диалоговое окно File > Import image (та же логика присутствует в *GIMP*). Когда вы это сделаете, другое диалоговое окно предложит вам выбрать нитки для вышивки, количество переплетений канвы на дюйм и общий масштаб рисунка. Импортированному изображению в ряде случаев требуется добавочное редактирование: вряд ли удастся употребить сорок моточков разных цветов для вышивки крошечных пикселей на настоящей ткани. Поэтому вам нужно будет упростить схему и отредактировать часть стежков вручную. *KXStitch* предлагает для этого широкий выбор специальных инструментов, в том числе отличный редактор палитры, инструменты для рисования и удаления рисунка, заполнение многоугольников и многое другое.

Это приложение — настоящий ветеран в мире FOSS. *KXStitch* появился в 2005 году как программа *Qt3*. С тех пор он разросся десятками расширений, а заодно перешел на *Qt4*, и ныне мы можем наслаждаться версией, которая предлагает больше переводов и несколько усовершенствований.

> «Предлагает широкий выбор специнструментов редактирования.»



› Оторвитесь от своего компьютера с Linux — повышивайте, что ли...

Уровень *KXStitch* для начинающих пользователей не так высок, как можно подумать. Фактически, здесь есть всестороннее руководство как по рисованию схемы с нуля, так и по обработке растровой графики, и все это объединено в удобный для обслуживания формат docbook.

Ссылка на скачивание *KXStitch* имеется на домашней странице приложений KDE. Это важно, потому что сайт *KXStitch* на Sourceforge давненько не обновлялся и предлагает устаревшую версию, тогда как страница kde-apps вполне новая. Компиляция *KXStitch* не представляет сложности благодаря превосходной оболочке *cmake*. Понадобятся заголовки рабочей среды KDE, *cmake* и стандартная среда компиляции, и после этого вам останется только запустить скрипт **./build.sh**.

Менеджер шифрованной файловой системы

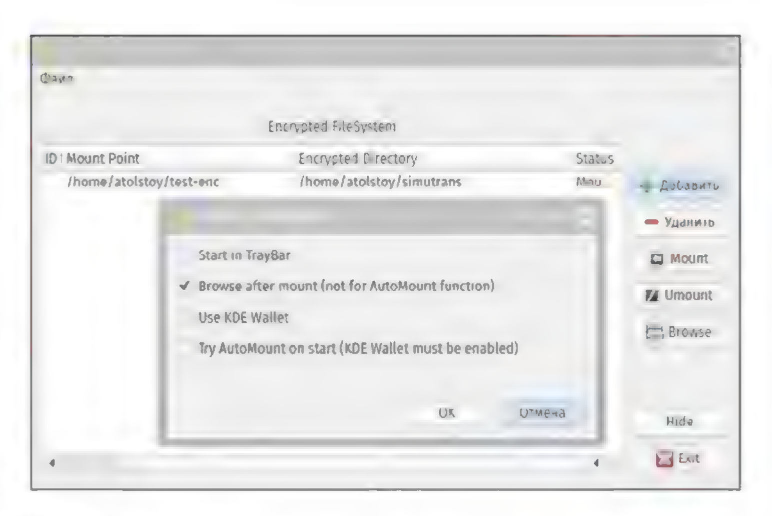
KEncFS

Версия: 1.4.0 **Сайт:** http://bit.ly/1pMtHOb

Обычно пользователей Linux считают людьми, озабоченными проблемами безопасности, хотя по большей части из-за стереотипного представления об альтернативной культуре и хакерах. Однако потребность в высококачественной технологии шифрования в этом пост-Сноуденовском мире уже выходит за рамки предприятий. Даже не очень опытные пользователи заинтересованы в том, чтобы хранить некоторые файлы в скрытом виде и в безопасности. Здесь-то и появляется *EncFS* — основанное на *FUSE* криптографическое решение, которое шифрует все в специальной директории. В отличие от программ шифрования диска типа *TrueCrypt*, *EncFS* работает на уровне директории, поэтому структура директории шифротекста отражает структуру директории простого текста. Иными словами, вы должны настроить для *EnFS* две директории: точку монтирования для вашего защищенного файла и зашифрованную директорию для шифрованных файлов. Каждый набор защищен главным паролем, который является единственным ключом к монтированию зашифрованного назначения. Трудно сказать, есть ли достоинства у попыток дешифровать шифротекст, но у *EncFS* хорошая репутация в среде профессионалов ИТ-безопасности.

KEncFS, вероятно, самый интуитивный и качественный интерфейс к *EncFS*, сконцентрированный на интеграции с KDE SC. У него опрятный и понятный интерфейс *Qt4* и кое-какие милые детали. Главное окно хранит список файловых систем, которые *KEncFS* использует для представления зашифрованных директорий. Пользователь может добавлять, удалять и управлять элементами, а также монтировать и размонтировать их. Когда элемент

> «KEncFS может автомонтировать шифрованные файловые системы.»



› Очень простой GUI для столь мощного решения по шифрования директории.

монтирован, автоматически открывается соответствующая директория в файловом менеджере по умолчанию (скажем, в *Dolphin*). Кроме того, *KEncFS* может автомонтировать ваши зашифрованные файловые системы при запуске и использовать *KWallet* для хранения паролей (не забудьте защитить сам *KWallet* сильным паролем), и мало влияет на производительность системы. *KEncFS* предлагает прекомпилированные бинарники, но есть шанс, что вам придется скомпилировать их самостоятельно. Поскольку это — довольно современный проект *Qt*, установка сводится к простым **qmake**, **make** и **make install**, хотя на установку всех необходимых зависимостей (в основном это библиотеки рабочей среды KDE) уходит некоторое время.

Свободный системный профайлер

I-Nex

Версия: 7.4.0 **Сайт:** http://bit.ly/1vMr4mp

Иногда нужно точно узнать разрешение своего экрана или модель чипсета материнской платы. А если так, то *I-Nex* — именно тот инструмент, который вы искали. Для тех, кто недавно пришел из мира Windows: он играет практически ту же роль, что и *CPU-Z*. Это профайлер (или монитор), который определяет CPU, ОЗУ, чипсет материнской платы и прочие функции оборудования вашего компьютера и представляет эту информацию в одном окне.

Окно *I-Nex* состоит из нескольких вкладок, и каждая посвящена некому аспекту системы. Вкладка CPU отображает полное имя вашего процессора, размеры кэша L1 и L2, текущую температуру и уровень нагрузки, и т. д. Во вкладке GPU легко найти спецификации графического чипа, узнать его способности насчет OpenGL и посмотреть подробную информацию по экрану (данные добываются из EDID). Полный список того, что отображает *I-Nex* во всех своих вкладках, очень велик, и это не лучший способ потратить место в журнале, но *I-Nex* — отличный товарищ для любителей оборудования и создателей систем. Другим пользователям он пригодится при определении внутренностей ноутбука или ПК.

Чтобы установить *I-Nex*, сначала проверьте свои системные репозитории Linux, потому что этот инструмент очень часто в них доступен. Если нет, скачайте tar-архив с сайта проекта. Вам не нужно ничего компилировать, потому что *I-Nex* распространяется как набор уже готовых файлов. Мы не говорим «бинарников», потому что *I-Nex* создан необычным способом: он требует наличия рабочей среды проекта Gambas (это диалект почтенного языка BASIC). *I-Nex* собирает информацию из десятков источников, и разумно запускать его из командной строки, отслеживая отсутствие важных элементов, например, датчиков температур.

> «Для тех, кто с Windows: он играет почти ту же роль, что и CPU-Z.»



› Полная спецификация вашей системы, идеально воссозданная.

HotGames Развлекательные приложения

Волейбол

Blobby Volley 2

Версия: 1.0rc1 **Сайт:** http://bit.ly/1nYrntA

Для нас *Blobby Volley 2* занимает особое место среди множества спортивных симуляторов. При первом знакомстве он вызывает смешанные чувства, и вы можете решить им пренебречь из-за его чрезмерной простоты. И напрасно. Внешность бывает обманчива, и через пару минут игры вы поймете, что *Blobby Volley 2* заслуживает гораздо большего внимания.

Вкратце, это классическая аркада для двух игроков, а чтобы несколько все запутать, недавняя версия игры — 1.0rc1, хотя на самом деле это римейк с открытым кодом старой проприетарной версии 1, с новым прозвищем: *Blobby Volley 2*.

Каждый игрок — это цветной неровный пузырь с простой анимацией движения, которая делает его похожей на вязкий гель. Однако игровой процесс *Blobby* мигом затягивает. Каждый игрок может использовать индивидуальные клавиши или мышь, что означает возможность удобно делить клавиатуру или дать возможность одному игроку использовать клавиатуру, а другому — мышь. *Blobby Volley 2* также позволяет играть с ботом (это, кстати, скрипт LUA). Конечно же, настоящее испытание — играть против своего друга: боты действуют слишком прямолинейно. Но для тренировки годятся.

Это отнюдь не все. Настоящее волшебство начинается, когда вы выходите в сеть и соединяетесь с одним из публичных серверов Blobby Volley League. Да, у этой игры очень сильное сообщество, с настоящими турнирами и чемпионатами, и все это размещается на blobby-volley.net. Можно сыграть со случайным игроком, или, что еще лучше, вызвать на поединок кого-нибудь из официальной лиги — и в том, и в другом случае обеспечена масса удовольствия.

> «В сети вы соединяетесь с одним из публичных серверов.»



› Попробуйте играть сразу за обе стороны и посмотрите, какая рука у вас выиграет.

Римейк Caesar III с открытым кодом

Caesaria

Версия: 0.4 **Сайт:** http://bit.ly/1ux8nmB

Иногда классика возвращается в еще большем блеске и славе, и сейчас наступила очередь *Caesar III* — очень популярной экономической игры с постройкой городов, выпущенной Sierra Entertainment в 1998 г.

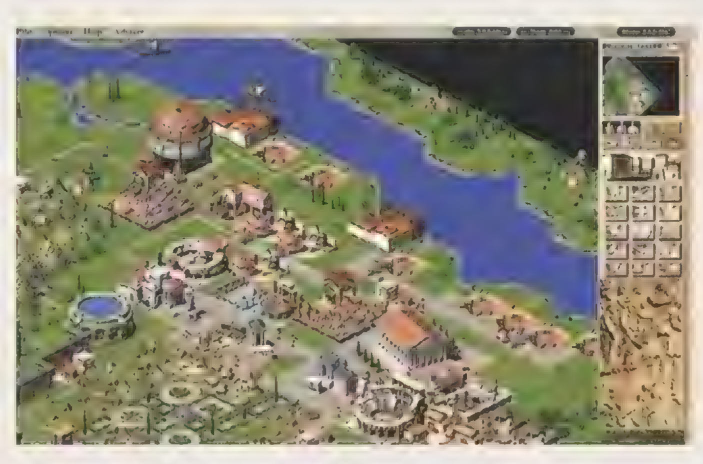
В 2013 г. хакер Linux по имени Dalerank затеял создание римейка с открытым кодом этой почтенной игры, причем большую часть кода написал с нуля. Игра стала очень похожим клоном оригинала, где тщательно воссоздавалась и постоянно расширялась его логика. Сайт проекта постоянно публикует новые сборки *Caesaria* (она же *OpenCaesar3*) для Windows, Linux и OS X, предлагая как 32-, так и 64-битные версии.

Для запуска игры просто распакуйте скачанный архив ZIP и запустите оттуда **./caesaria.linux**. Игра зависит только от набора библиотек SDL, и если какой-то из них не хватает, вы сразу выясните, которой именно, по информации в командной строке.

Игра попросит вас поместить некоторые оригинальные ресурсные файлы *Caesar III* в определенные директории для аудиоподдержки и дополнительных карт местности, но даже если у вас этого нет, простая сборка *Caesaria* уже содержит всю графику, вполне достаточную, чтобы начать строить римский город.

Цель игры — добыть себе желаемый ранг, достигая заданных Императором целей по строительству и развитию города. Игра довольно точно отражает жизнь древних римлян — например, их жилища, производство товаров и торговлю, борьбу с варварами, организацию развлечений и жертвоприношения богам. Dalerank написал в своем блоге, что сложнее всего было написать математические условия для расчета маршрутов и поведения граждан. Хотя пока следует признать, что в Caesaria нужно поработать над стабильностью (она периодически слетает при попытках повернуть карту), все же в игру вполне можно играть, и она сама по себе является большим достижением.

> «Цель — добыть ранг, достигая заданных Императором целей...»



› Возможно, стоит понастроить больше гончарен — не пропадать же всей этой глине. Да, вот такие мы Плюшкины.

Web-браузер

Otter Browser

Версия: 0.9.03 **Сайт:** http://otter-browser.org

Otter Browser (не путать с ПО автоматического доказательства теорем *Otter*) — это смелый проект *Qt5*, чья цель — воссоздание интерфейса и функций *Opera 12.x*. Как вы, вероятно, знаете, в 2012 г. Opera Software прекратила поддержку своей ключевой технологии, движка рендеринга Presto HTML, который являлся сердцевиной как настольного, так и мобильного браузера Opera, а также других продуктов. В то время команда разработки Opera Software тоже сократилась, и компания решила перенести свои усилия на проект *Chromium* и помочь Google разработать код для их браузера с открытым кодом. С тех пор «новая» *Opera* использует движок Blink от Google и приобрела внешнее сходство с *Chromium*. Фактически, в *Opera* были выброшены самые любимые функции Presto — ради того, чтобы приспособиться к переделанному клону *Chromium*. Эти функции составили солидный список, и среди них — менеджер закладок, встроенная программа блокировки контента, переключатель JavaScript, переключатель отображения картинок, отложенная загрузка вкладок, распознавание жестов мыши, умное выделение текста и многое, многое другое.

Люди из *Otter Browser* намерены восстановить все эти выдающиеся функции, не теряя индивидуальности классической *Opera*. Проект находится на стадии бета и предлагает дополнительную поддержку расширения Adblock Plus, поддержку импорта закладок HTML, индивидуальную настройку клавиш быстрого запуска, менеджер загрузок, менеджер cookie, удобную боковую панель и множество второстепенных улучшений пользовательского интерфейса. Кроме того, *Otter Browser*, похоже, единственный (или почти) браузер, основанный на виджетах *Qt5*, и если вы планируете обзавестись рабочим столом чисто на *Qt5*, например, *Lumina*, то *Otter Browser* идеально туда впишется.

> «Он создан, чтобы воссоздать классические функции Opera.»



› **В стандартном тесте HTML5 браузер демонстрирует весьма приличные результаты. Неплохо для бета-версии.**

По части установки, *Otter Browser* вошел в пакеты для разных вариантов Linux, в том числе openSUSE, Ubuntu, Arch и их производных. Однако, поскольку проект разрабатывается довольно быстро (разработчики публикуют отчеты на своем сайте еженедельно), было бы неплохо самим скомпилировать самую свежую версию из Git. Понадобятся свежие версии *cmake* и пакета разработки *Qt5* с заголовками *QtWebkit 5.x*.

Менеджер мобильных устройств и соединений

ModemManager

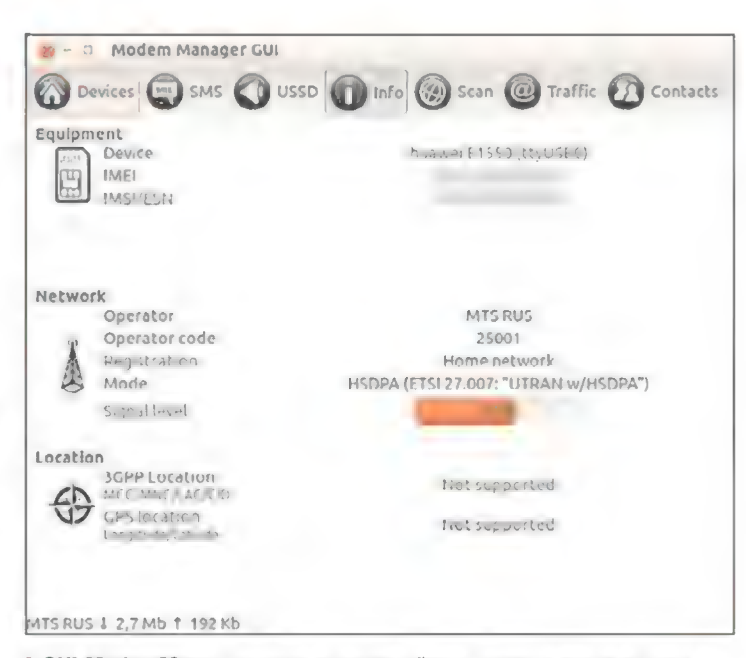
Версия: 1.4 **Сайт:** http://bit.ly/1pLSpzy

Несмотря на развитие мобильных технологий, мы иногда по-прежнему используем модемы, особенно на ходу или в тех местах, где нет проводного интернета. Модемы скрываются практически в любом телефоне и в любом смартфоне, не говоря уж про устройства USB. Прошли те дни, когда мы учили команду *ifup* и возились с аргументами *ppp* в Linux, и *ModemManager* отразил эти изменения — это *NetworkManager*-подобный инструмент для работы с модемами, периферийными устройствами, телефонами Bluetooth и даже более сложными профессиональными устройствами со внешним питанием.

ModemManager — это демон DBus; он контролирует мобильные (2G/3G/4G) устройства и соединения и предоставляет функции для управления звонками и SMS, а также контактную информацию и некоторые функции GSM. Если вам требуется интернет-соединение на компьютере через смартфон, применяемый как модем (при наличии соответствующего тарифного плана на телефоне), а также соединение Bluetooth или USB, *ModemManager* предложит вам более разумный и современный вариант действий, чем Bluedevil+KPPP или *wvdial* для телефонного модема.

ModemManager предлагает GUI, запускаемый так же, как любое отдельное приложение, и автоматически вызываемый при соединении вашего телефона через Bluetooth, если вы решили добавить опцию DUN (dial up networking). Таким образом соединение добавляется в настройку *NetworkManager* и отображается среди других доступных соединений там.

> «Инструмент для работы со встроенными модемами и т. п.»



› **GUI *ModemManager* — это настоящий пульт управления вашим модемом в Linux.**

Большинство производителей Linux считают *ModemManager* совершенно обязательной программой, поэтому вы наверняка найдете свежий релиз в постоянных обновлениях. Новая версия 1.4 умеет управлять энергоснабжением устройства, лучше работает с IP (например, вы можете настроить статический IPv6-адрес вместе с режимом DHCP) и поддерживает современные модемы ZTE, Huawei и Sequans Communcation. LXF

На диске

Дистрибутивы, приложения, игры, подкасты и всякое-разное...

Лучшее из Интернета, упакованное в 9 ГБ качественного DVD.



Дистрибутивы

На DVD этого месяца царит Raspberry Pi, и этот крошечный компьютер, несомненно, захватил сцену Linux, да и компьютерных технологий в целом, буквально с момента своего появления в начале 2012 г. Вам простительно думать, что это единственный компьютер подобного рода, но это не так. Например, Beaglebone Black мощнее, хотя на £10 дороже — и вы вряд ли слышали о нем. Разумеется, рекламная кампания, предварившая появление Pi, способствовала изначальному интересу к нему, но это не объясняет сохранения интереса почти три года спустя.

Фактор, приведший к успеху Raspberry Pi — тот же, что привел к растущему успеху Linux в целом, и это — сообщество. Можно что угодно говорить о спецификациях оборудования, но в первую голову значение имеют люди. Это они выдвигают самые невероятные идеи проектов на Pi, и они же позволяют маленьким компаниям расти — потому что не покупай они продукцию этих компаний, таковых попросту не существовало бы. Успех Pi отражает то, что движет Linux и открытым кодом, просто он заметнее: ведь перед вами эта маленькая плата, на которой вы можете сосредоточиться.

Neil

» Важно ВНИМАНИЕ!

Порченые диски

В маловероятном случае какого-то дефекта вашего **LXFDVD** обращайтесь, пожалуйста, по адресу disks@linuxformat.ru или телефону +7 (812) 309-0686.

Менеджер установки ОС

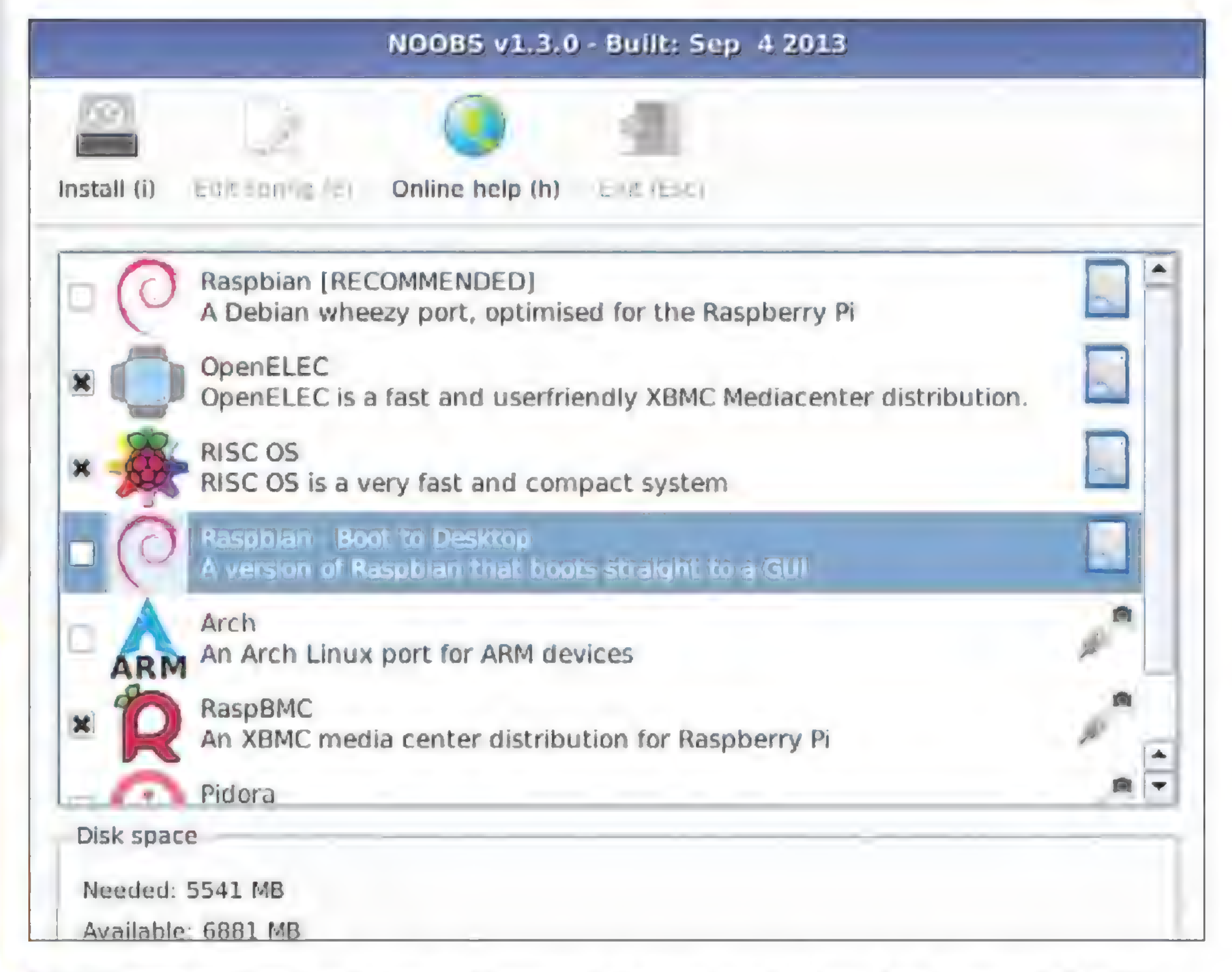
NOOBS 1.3.10

Когда мы говорим, что на **LXFDVD** этого месяца у нас пять дистрибутивов Raspberry Pi, это не совсем так. На самом деле их четыре, да еще *NOOBS*. *NOOBS* (New Out Of Box Software) обходит проблемы, связанные с традиционным методом копирования дистрибутива Pi на карту SD командой **dd**: требование прав root или риск указать неверное имя диска (**dd** не склонен прощать ошибки). В *NOOBS* вы просто распаковываете ZIP-архив на SD-карту, как обычный пользователь, и затем загружаете Pi с карты.

Так что же такое *NOOBS*, если не дистрибутив? А это подборка шести разных дистрибутивов плюс установщик: загрузитесь с SD-карты, и вы увидите список. Среди них стандартный Raspbian с портами других дистрибутивов, знакомых пользователям настольных ПК: например, Arch и Fedora. Здесь также есть спецдистрибутивы для медиа-центров, OpenELEC и RaspBMC, и, для разнообразия, порт RiscOS.

Выберите желаемые в меню, и *NOOBS* установит их. После перезагрузки *NOOBS* представит вам меню, где выбирается дистрибутив, если вы установили более одного, точно так же, как при двойной загрузке с *Grub*... только симпатичнее. И что еще лучше, *NOOBS* останется на SD-карте, и доступ к нему можно будет получить позже, удерживая клавишу Shift при загрузке. Вы попадете в интерфейс аварийного восстановления, где можно вылечить поврежденную систему или поставить другой дистрибутив. Учтите, что некоторые дистрибутивы требуют наличия интернет-соединения; такие отмечены значком справа от имени.

NOOBS означает, что вам больше не надо перебирать SD-карты для перехода на другой дистрибутив — на картах редко бывают ярлычки, и их часто путают, особенно если у вас модель Pi B+, применяющая карты microSD: отметки на них не сделаешь, и их почти невозможно не потерять. На одной карте содержится все необходимое, и кроме того, в *NOOBS* есть опция резервирования раздела для данных с разделенным доступом. Хотя сам *NOOBS* занимает на вашей SD-карте всего 700 МБ, мы советуем использовать карту не менее чем на 4 ГБ, а желательно 8 ГБ, чтобы хватало места для установленных дистрибутивов.



› *NOOBS* предлагает шесть дистрибутивов и позволяет установить один или более без обращения к dd или CLI.



Новичок в Linux? Начните отсюда!

» Что такое Linux? Как его установить?
» Есть ли в нем эквивалент *MS Office*?
» Зачем нужна командная строка?
» Как устанавливать программы?

Ответы приводятся в Index.html на диске.

KaliLinux, Jasper & RetroPie

Новые радости Pi

Другие дистрибутивы на DVD демонстрируют многогранность Raspberry Pi. Помимо общецелевых настольных сред в *NOOBS*, у нас есть медиа-серверы, «музыкальный автомат [jukebox]», дистрибутив для тестирования возможности проникновений и платформа для ретро-игр. Помимо *NOOBS*, остальные предлагаются в виде сжатых образов диска. Распакуйте TAR или ZIP, и вы обнаружите файл **.img**, который нужно скопировать на SD-карту. Убедитесь, что вы правильно назначили устройство, не то придется распроститься с данными на своем жестком диске.

Затем скопируйте образ:

```
sudo dd if=/path/to/image of=/dev/sdX bs=4k
```



Музыкальный сервер

PiMusicBox 0.5

В этом месяце Сравнение посвящено медиа-плейерам, и одним из кандидатов был *Music Player Daemon* (*MPD*). Оказалось, *MPD* также и основа для этого дистрибутива Pi. PiMusicBox использует Mopidy (www.mopidy.com), который, в свою очередь, основан на *MPD*, и каждый уровень расширяет и без того богатый набор функций *MPD*. Это музыкальный сервер, он сидит где-то обособленно (Pi уж явно обособленный), соединенный с вашей аудиосистемой и сетью. Им можно управлять прямо из браузера или через множество клиентов *MPD*, доступных для настольного Linux и Android.

PiMusicBox воспроизводит музыку из присоединенного локального хранилища или из любого места вашей сети, но это далеко не все, что он умеет делать. PiMusicBox также воспроизводит потоковое аудио с многих популярных онлайн-серверов, включая Spotify, SoundCloud, Google Music, Subsonic и Soma FM. Он также воспроизводит подкасты с gPodder и iTunes. Все управляется из web-браузера или из удаленных клиентов — чтобы что-либо настроить, лезть в командную строку нужды нет; разве что для подключения беспроводной сети. LXF

И еще!

Системные инструменты

Главное

Checkinstall Установка tar-архива с помощью менеджера пакетов.

GNU Core Utils Основные утилиты, обязанные присутствовать в каждой операционной системе.

Hardinfo Инструмент для тестирования системы.

Ядро Исходник последней версии ядра.

Memtest86+ Проверьте свое ОЗУ на сбои и ошибки.

Plop Простой менеджер загрузки для запуска операционных систем.

RaWrite Создавайте загрузочные дискеты в Windows.

SBM Независимый от ОС менеджер загрузки с простым в использовании интерфейсом.

WvDial Соединяйтесь с Интернетом через телефонный модем.

Чтение

Книжная полка

The Cathedral and the Bazaar [Собор и Базар] Классический текст Эрика С. Реймонда [Eric S Raymond], объясняющий преимущества открытой разработки.

Linux Kernel in a Nutshell [Ядро Linux в двух словах] Введение в ядро, написанное магистром хакеров Грегом Кроа-Хартманом [Greg Kroah-Hartman].

Справочник администратора Debian Руководство для системных администраторов.

Словарь Linux Linux от A до Z.

Окунитесь в Python Мастер-класс по этому популярному языку.

Руководство по созданию скриптов Bourne Shell Начинайте осваивать скрипты оболочки.

Расширенное руководство по созданию скриптов Bash Больше информации для опытных пользователей.

Intro to Linux Руководство для новичков.

Руководство сисадмина Основы работы с системой.

Обзор утилит Демонстрация инструментария GNU.



Пропустили номер?

Закажите его через сайт www.linuxformat.ru в «ГНУ/Линуксцентре»! Журналы доставляются и в печатной, и в электронной форме, так что с момента открытия браузера до получения нужного вам выпуска **LXF** может пройти всего пара минут!

Прямо сейчас для заказа доступны следующие номера:



LXF188
Октябрь 2014

- **Разгоним Linux** Со скоростью мира — самое время
- **Распознавание речи** Слушать мою команду
- **Этот логичный Arch** Летучий и катучий
- **Ядрен корень** Модифицируем ядро Linux от имени root

LXFDVD: Arch Linux 2014_7, CentOS 7.0, Netrunner 14, Peppermint Five, Rescatux 0.32, 10 книг о Linux (на английском языке), горячие новинки и прочее...

Печатная версия: shop.linuxformat.ru/lxf_188/
PDF-версия: shop.linuxformat.ru/elxf_188/



LXF189
Ноябрь 2014

- **Нет проблем!** Или — решим легко
- **Ваш портативный друг** Выбираем Chromebook
- **Прошлое криптографии** Как человечество хранило секреты
- **Осваиваем макросы** Автоматизация офиса

LXFDVD: Tails 1.1 Live OS, Deepin 2014, Rescatux 0.32b1, System Rescue CD 4.3.0, Ultimate Boot CD 5.3.0, 10 книг о Linux (на английском языке), горячие новинки и прочее...

Печатная версия: shop.linuxformat.ru/lxf_189/
PDF-версия: shop.linuxformat.ru/elxf_189/



LXF190
Декабрь 2014

- **Кто на свете всех милее?** Дистрибутив-2014
- **Жизнь в Сети** Браузер вашей мечты
- **PhotoRec** Чьих рук дело — спасение файлов
- **Pi и Mathematica** Две знаменитости

LXFDVD: Linux Mint Debian Edition, Trisquel, Mageia, CentOS, Korora, openSUSE, Salix, 10 книг о Linux (на английском языке), горячие новинки и прочее...

Печатная версия: shop.linuxformat.ru/lxf_190/
PDF-версия: shop.linuxformat.ru/elxf_190/

А чтобы не упустить ничего, оформите подписку! Все, кто подписался на печатную версию журнала через www.linuxformat.ru/subscribe/ или www.linuxcenter.ru, получают электронную версию в подарок! На сайте shop.linuxformat.ru вы также сможете приобрести предыдущие выпуски **LXF**.

Подписывайтесь на сайте www.linuxformat.ru/subscribe

Телефоны отдела подписки:
- Санкт-Петербург (812) 309-0686
- Москва (499) 271-4954



На странице LXF ВКонтакте вы найдете:

- Новости о Linux
- Статьи из архива **LXF**
- Анонс свежего выпуска **LXF** и часть статей из него
- Живое общение и онлайн-консультацию по подписке



Информация о диске

Что-то потеряли?

Часто случается, что новые программы зависят от других программных продуктов, которые могут не входить в текущую версию вашего дистрибутива Linux.

Внимательно прочтите это перед тем, как использовать LXFDVD!

Мы стараемся предоставить вам как можно больше важных вспомогательных файлов. В большинстве случаев, последние версии библиотек и другие пакеты мы включаем в каталог «Essentials [Главное]» на прилагаемом диске. Поэтому, если в вашей системе возникли проблемы с зависимостями, первым делом следует заглянуть именно туда.

Форматы пакетов

Мы стараемся включать как можно больше различных типов установочных пакетов: RPM, Deb или любых других. Просим вас принять во внимание, что мы ограничены свободным пространством и доступными двоичными выпусками программ. По возможности, мы будем включать исходные тексты для любого пакета, чтобы вы могли собрать его самостоятельно.

Документация

На диске вы сможете найти всю необходимую информацию о том, как устанавливать и использовать некоторые программы. Пожалуйста, не забывайте, что большинство программ поставляются вместе со своей документацией, поэтому дополнительные материалы и файлы находятся в соответствующих директориях.

Что это за файлы?

Если вы новичок в Linux, вас может смутить изобилие различных файлов и расширений. Так как мы стараемся собрать как можно больше вариантов пакетов для обеспечения совместимости, в одном каталоге часто находятся два или три файла для различных версий Linux и различных архитектур, исходные тексты и откомпилированные пакеты. Чтобы определить, какой именно файл вам нужен, необходимо обратить внимание на его имя или расширение:

» **имя_программы-1.0.1.i386.rpm** — вероятно, это двоичный пакет RPM, предназначенный для работы на системах x86;
» **имя_программы-1.0.1.i386.deb** — такой же пакет, но уже для Debian;
» **имя_программы-1.0.1.tar.gz** — обычно это исходный код;
» **имя_программы-1.0.1.tgz** — тот же файл, что и выше этажом по списку: "tgz" — это сокращение от "tar.gz";
» **имя_программы-1.0.1.tar.bz2** — тот же файл, но сжатый bzip2 вместо обычного gzip;
» **имя_программы-1.0.1.src.rpm** — также исходный код, но поставляемый как RPM-пакет для упрощения процесса установки;
» **имя_программы-1.0.1.i386.FC4.RPM** — двоичный пакет RPM для x86, предназначенный специально для операционной системы Fedora Core 4;
» **имя_программы-1.0.1.ppc.Suse9.rpm** — двоичный пакет RPM, предназначенный специально для операционной системы SUSE 9.x PPC;
» **имя_программы-devel-1.0.1.i386.rpm** — версия для разработчиков.

Если диск не читается...

Это маловероятно, но если все же прилагаемый к журналу диск поврежден, пожалуйста, свяжитесь с нашей службой поддержки по электронной почте: disks@linuxformat.ru

LINUX FORMAT DVD Главное в мире Linux

ДИСТРИБУТИВЫ | ПО | УЧЕБНИКИ

Для вашего Pi

- Raspbian
- KaliLinux 1.0.9
- PiMusicBox 0.5
- RetroPie 2.3
- Jasper 2014

LXF191/192 | Откройте index.html — и за исследование!



» Содержание

LINUX FORMAT

Пожалуйста, перед использованием данного диска ознакомьтесь с инструкцией, опубликованной в журнале на стр. 109!

КОММЕНТАРИИ? Присылайте ваши пожелания и предложения по электронной почте info@linuxformat.ru

ДЕФЕКТНЫЕ ДИСКИ В маловероятном случае обнаружения дефектов на данном диске, обращайтесь, пожалуйста, по адресу disks@linuxformat.ru

Тираж изготовлен ООО «Маркон», 188652, Россия, Ленинградская область, Всеволожский р-н, дер. Юкки, Школьная ул., 7 а. Лицензия МПТР ВАФ N 77-103

Создание установочных дисков при помощи cdrecord

Самый быстрый способ записать ISO-образ на чистую матрицу — это обратиться к программе *cdrecord*. Для всех перечисленных ниже действий потребуются права суперпользователя-root. Сначала определите путь к вашему устройству для записи дисков. Наберите следующую команду:

```
cdrecord -scanbus
```

После этого на экране терминала должен отобразиться список устройств, подключенных к вашей системе. SCSI-адрес каждого устройства представляет собой три числа в левой колонке — например, 0,3,0. Теперь вы можете с легкостью записать образ на диск:

```
cdrecord dev=0,3,0 -v /путь к образу/image.iso
```

Чтобы упростить дальнейшее использование *cdrecord*, сохраните некоторые настройки в файле **/etc/default/cdrecord**. Добавьте по одной строке для каждого устройства записи (скорее всего, в вашей системе присутствует только одно такое устройство):

```
Plextor= 0,3,0 12 16M
```

Первое слово в этой строке — метка; затем после адреса SCSI-устройства вы должны указать скорость и размер буфера. Теперь можете заменить SCSI-адрес в командной строке на выбранную вами метку. Все будет еще проще, если вы добавите следующее:

```
CDR_DEVICE=Plextor
```

Для записи ISO-образа вам осталось набрать команду

```
cdrecord -v /path/to/image.iso
```

Если вы не принадлежите к любителям командной строки, в таком случае вам придет на помощь утилита *gcombust*. Запустите ее от имени root и выберите вкладку Burn и ISO 9660 Image в верхней части окна. Введите путь к образу, который вы хотите записать на диск, и смело нажимайте на Combust! Пока ваш образ пишется на диск, можете выпить чашечку кофе.

Другая ОС?

Использовать Linux для записи компакт-диска не обязательно. Программы вроде *cdrecord* просто переносят двоичные данные на чистую матрицу. Все необходимые файлы уже включены в ISO-образ, который распознается любой операционной системой, будь то Linux, Windows, Mac OS X или AmigaOS.

Нет устройства для записи дисков?

Если у вас нет устройства, с помощью которого можно было бы записать образ на диск, можно найти какого-нибудь друга или организацию, у кого есть компьютер с дисководом, и прожечь диск у них. Опять-таки, вам подойдет любая операционная система, способная распознать пишущий привод (см. выше).

Некоторые дистрибутивы умеют монтировать образы дисков и выполнять сетевую установку или даже установку с раздела жесткого диска. Конкретные методы, конечно, зависят от дистрибутива. За дополнительной информацией обращайтесь на web-сайт разработчика дистрибутива.



В февральском номере

Лучшие проекты для Linux

Создайте лучший в мире ящик для потокового вещания фильмов и музыки, показа фоток и хранения ТВ-программ.

RAID значит «наезд»

Погрузимся в мир RAID в поисках лучшей файловой системы и перестроимся за рекордный срок.

Ядро Minix

Ужас! На страницы **LXF** проникло чужеродное ядро. Рассматриваем Minix и что с ним стряслось.

Minetest

Теперь *Minecraft* принадлежит Microsoft, а мы со своими блочными мирами уносимся в истинно открытую гавань.

Содержание будущих выпусков может меняться — вдруг нас повяжут за политику безопасности...



Журнал зарегистрирован Федеральной службой по надзору за соблюдением законодательства в сфере массовых коммуникаций и охране культурного наследия ПИ № ФС77-21973 от 14 сентября 2005 года. Выходит ежемесячно. Тираж печатной версии 3000 экз., распространение электронной версии 30000 экз.

РЕДАКЦИЯ РУССКОЯЗЫЧНОЙ ВЕРСИИ

Главный редактор
Кирилл Степанов info@linuxformat.ru
Литературный и выпускающий редактор
Елена Толстякова
Переводчики
Максим Алибаев, Елена Ессяк, Даниил Кривошеин, Светлана Кривошеина, Валентин Развозжаев, Елена Толстякова
Редактор диска
Александр Баракин
Верстка, допечатная подготовка
Сергей Рогожников
Технический директор
Андрей Смирнов
Директор по рекламе
Владимир Савельев advert@linuxformat.ru
Генеральный директор
Павел Фролов
Учредители
Частные лица
Издатель
ООО «Линукс Формат»

Отпечатано в типографии ООО «ЛД-ПРИНТ»
196644, Санкт-Петербург, Колпинский р-н, пос. Саперный, территория предприятия «Балтика», д. б/н, лит. Ф
Тел. (812) 462-8383, e-mail: office@ldprint.ru
Заказ 11998

РЕДАКЦИЯ АНГЛОЯЗЫЧНОЙ ВЕРСИИ

Редактор Нейл Мор [Neil Mohr] neil.mohr@futurenet.com
Научный редактор Джонни Бидвелл [Jonni Bidwell] jonni.bidwell@futurenet.com
Выпускающий редактор Крис Торнетт [Chris Thornett] chris.thornett@futurenet.com
Художественный редактор Эфраин Эрнандес-Мендоса [Efrain Hernandez-Mendoza] efrain.hernandez-mendoza@futurenet.com

ПОДГОТОВКА МАТЕРИАЛОВ

Джонни Бидвелл [Jonni Bidwell], Нейл Ботвик [Neil Bothwick], Крис Браун [Chris Brown], Джолион Браун [Jolyon Brown], Ричард Коббетт [Richard Cobbett], Шон Конвэй [Sean Conway], Кент Ельчук [Kent Elchuk], Мэтт Хэнсон [Matt Hanson], Филип Херрон [Philip Herron], Лес Паундер [Les Pounder], Маянк Шарма [Mayank Sharma], Шашанк Шарма [Shashank Sharma], Ричард Смедли [Richard Smedley], Александр Толстой [Alexander Tolstoy], Михалис Цукалос [Mihalis Tsoukalos], Евгений Балдин, Андрей Гондаренков, Алексей Федорчук

Поддержка выпуска Алекс Саммерсби [Alex Summersby], Ян Ивенден [Ian Evenden]
Иллюстрации Шейн Коллиндж [Shane Collinge], Саймон Миддлвик [Simon Middleweek]

КОНТАКТНАЯ ИНФОРМАЦИЯ

UK: Linux Format, 30 Monmouth Street, Bath BA1 2BW
Тел. +44 01225 442244, email: linuxformat@futurenet.com
РОССИЯ:
Санкт-Петербург (редакция):
пр. Медиков, 5, корп. 7. Тел. +7 (812) 309-0686
По вопросам сотрудничества, партнерства, оптовых закупок: partner@linuxcenter.ru

Новое поколение средств защиты

Межсетевые экраны ССПТ, не имеющие IP-адреса

ССПТ-2 — это сертифицированное ФСТЭК, ФСБ и ГАЗПРОМСЕРТ средство защиты информации нового поколения, реализующее функции межсетевого экрана, но при этом остающееся «невидимым» для любых протоколов и тестовых воздействий, что достигается за счет отсутствия физических и логических адресов на его фильтрующих интерфейсах. **ССПТ-2 невозможно обнаружить никакими известными средствами удаленного мониторинга сети.**

Скрытность функционирования межсетевого экрана повышает надежность системы защиты в целом и существенно упрощает процедуру установки ССПТ-2 в компьютерные сети и функционирующие на их основе информационные и телематические системы.

Защита для высокоскоростных корпоративных сетей Ethernet 100/1000 Мбит/с

Сертифицированы ФСТЭК и ФСБ (3-й класс защиты)

На базе процессоров с 64-разрядной многоядерной архитектурой

FNP FIREWALL NETWORK PROCESSOR

Console 1 2 3 4

PWR HDD

Назначение устройства

Основное средство защиты для реализации различных политик информационной безопасности с помощью:

- фильтрации пакетов на канальном, сетевом, транспортном и прикладном уровнях;
- управления транспортными соединениями между отдельными узлами ЛВС или виртуальной ЛВС (VLAN);
- контроля контента данных на прикладном уровне с учетом направления, времени и типа протоколов передачи трафика.

Дополнительное устройство защиты для:

- обеспечения безопасности функционирования ранее установленных в компьютерной сети средств защиты и устройств маршрутизации;
- мониторинга трафика с возможностью анализа данных регистрации пакетов по различным критериям и интеграции с IDS;
- обеспечения функционирования сетевых распределенных телематических приложений и GRID-ресурсов.





Москва +7 (499) 271-49-54

Санкт-Петербург +7 (812) 309-06-86



Linux-эксперт для вашего бизнеса. www.linuxcenter.ru

HETZNER DEDICATED SERVERS



В ЦЕЛОСТИ И СОХРАННОСТИ!

ХРАНЕНИЕ ДАННЫХ В ГЕРМАНИИ



- Intel®Xeon® E3-1270 v3 Quadcore Haswell
- 32 ГБ ECC RAM
- 2 x 2 ТБ 6 Гбит/с SATA 7200 об/м Enterprise класс Software-RAID 1
- 20 ТБ Трафик*
- Операционная система Linux
- Подключение 1 Гбит порт
- Гарантировано 200 Мбит/с
- IPv6 подсеть (/64)
- Без минимального контракта

2600 рублей в месяц
\+ Установка 3700 рублей



ВЫДЕЛЕННЫЙ СЕРВЕР
PX70-SSD

- Intel®Xeon® E3-1270 v3 Quadcore Haswell
- 32 ГБ ECC RAM
- 2 x 480 ГБ 6 Гбит/с SATA SSD Data Center Series
- 30 ТБ Трафик*
- Операционная система Linux
- Подключение 1 Гбит порт
- Гарантировано 200 Мбит/с
- IPv6 подсеть (/64)
- Без минимального контракта

3700 рублей в месяц
\+ Установка 3700 рублей



* Нет платы за превышение. При превышении 20 ТБ/месяц (PX60), 30 ТБ/месяц (PX70-SSD) скорость соединения ограничивается (подсчёт ведётся по исходящему трафику, входящий и внутренний трафик не учитывается).

Безопасное защищённое хранение ваших данных в собственных дата-центрах немецкой компании Hetzner Online, соответствующее высоким требованиям к стандартам защиты данных в Германии.

RU.HETZNER.COM









Hetzner Online активно поддерживает защиту окружающей среды, используя исключительно 100% возобновляемые источники энергии. Выбирайте